

netkit lab

EBTables in bridged VLAN environment

Version	1.0
Author(s)	Sandro Doro
E-mail	sandro.doro@istruzione.it
Web	http://www.tic.fdns.net/tic/html/lab.html
Description	Simple example in using EBTable in a bridge VLAN network.

copyright notice

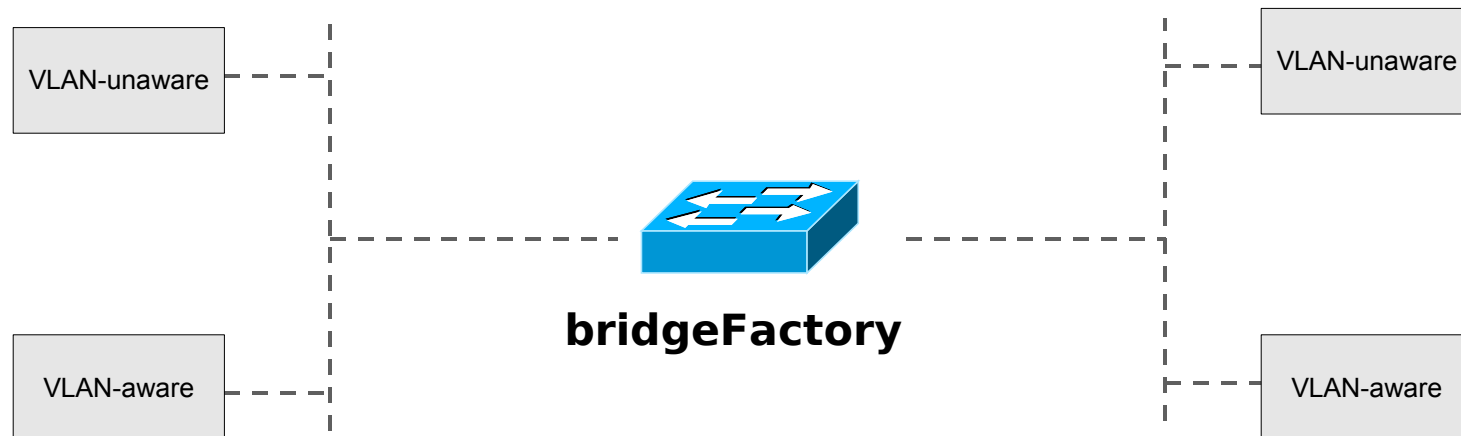
- All the pages/slides in this presentation, including but not limited to, images, photos, animations, videos, sounds, music, and text (hereby referred to as “material”) are protected by copyright.
- This material, with the exception of some multimedia elements licensed by other organizations, is property of the authors and/or organizations appearing in the first slide.
- This material, or its parts, can be reproduced and used for didactical purposes within universities and schools, provided that this happens for non-profit purposes.
- Information contained in this material cannot be used within network design projects or other products of any kind.
- Any other use is prohibited, unless explicitly authorized by the authors on the basis of an explicit agreement.
- The authors assume no responsibility about this material and provide this material “as is”, with no implicit or explicit warranty about the correctness and completeness of its contents, which may be subject to changes.
- This copyright notice must always be redistributed together with the material, or its portions.

Separate traffic in hybrid link

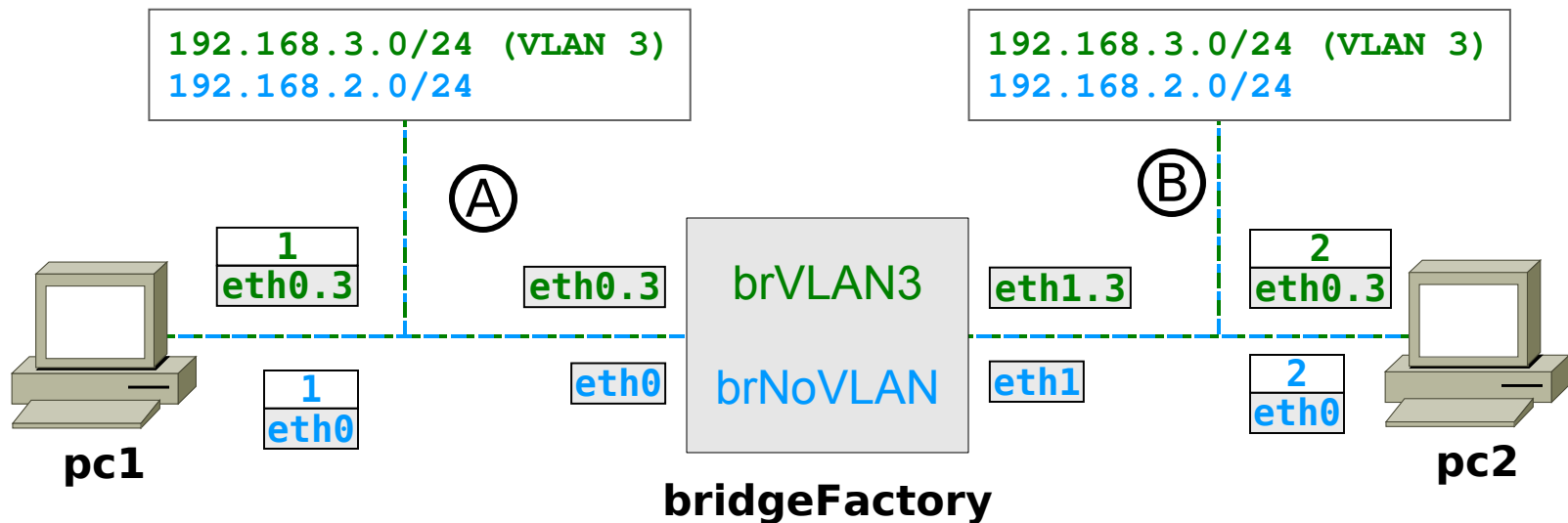
Assignment:

From two hybrid link (tagged and untagged) connected to a bridgeFactory node with two interfaces (eth0 and eth1), build two bridges: one for a VLAN (e.g #3) and the other for all untagged traffic.

abstract topology



detailed topology



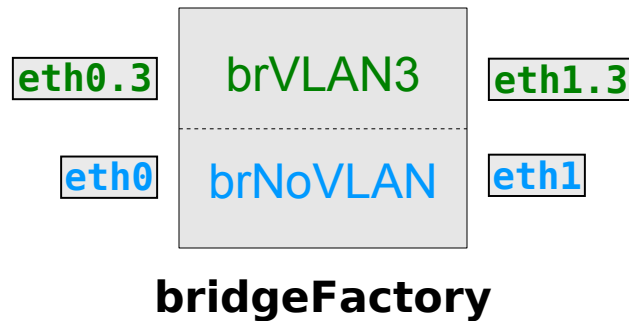
```
# brctl show
bridge name      bridge id        STP enabled  interfaces
brNoVLAN         8000.000000010000  no          eth0
                                                         eth1
brVLAN3          8000.000000010000  no          eth0.3
                                                         eth1.3
```

Problem



**sniffing
brNoVLAN**

The traffic go all trough brNoVLAN bridge



Solution: EBTables

Kernel space

Integrated into kernel 2.6 and patchable into kernel 2.4 (LEAF/Bering ready)

Enable in “Bridge: Netfilter Configuration” section:

```
CONFIG_BRIDGE_NT_EBTABLES=m  
CONFIG_BRIDGE_EBT_* =m
```

User space: <http://ebtables.sourceforge.net/>

arptables: is used to set up, maintain, and inspect the tables of
ARP rules in the Linux kernel

ebtables: is used to set up, maintain, and inspect the tables of
Ethernet frame rules in the Linux kernel.

Solution: EBTables

-t broute, is used to make a brouter, it has one built-in chain: BROUTING.

The targets DROP and ACCEPT have special meaning in the broute table. DROP actually means the frame has to be routed, while ACCEPT means the frame has to be bridged.

The BROUTING chain is traversed very early. It is only traversed by frames entering on a bridge enslaved NIC that is in forwarding state. Normally those frames would be bridged, but you can decide otherwise here.

EBTables commands

Filtering:

```
ebtables -t broute -A BROUTING -i eth0 -p 802_1q --vlan-id 3 -j DROP  
ebtables -t broute -A BROUTING -i eth1 -p 802_1q --vlan-id 3 -j DROP
```

Counting purpose:

```
ebtables -t broute -A BROUTING -i eth0.3 -p ipv4 -j CONTINUE  
ebtables -t broute -A BROUTING -i eth1.3 -p ipv4 -j CONTINUE
```



Logging:

```
ebtables -t broute -A BROUTING --log-ip --log-arp
```

Display counter:

```
ebtables -t broute -L BROUTING --Lc
```

modules loaded

 **bridgeFactory** 

```
# lsmod
Module                Size  Used by
ebt_log               3040   1
ebt_vlan              2788   2
ebtable_broute        1568   1
ebtables              20576   3 ebt_log,ebt_vlan,ebtable_broute
bridge                45720   1 ebtable_broute
atm                   40824   1 bridge
8021q                 17544   0
```

ebtables: base module for ebtables
ebtable_broute: bridge/routing decisions
ebtable_vlan: 802.1Q vlan match/filtering
ebtable_log: logging to the syslog

Testing brNoVLAN

▼ **bridgeFactory**

```
# tcpdump -i brNoVLAN -n
```

▼ **pc1**

```
# ping -c 1 192.168.2.2
```

Testing brVLAN3

▼ **bridgeFactory**

```
# tcpdump -i brVLA3N -n
```

▼ **pc1**

```
# ping -c 1 192.168.3.2
```