Gruppo Security - Topologie di rete

Alfieri, Belluomo, Carbone, Covati

- Programma:
  - analisi preliminare dei rischi;
  - definizione di strumenti e topologie standard che coprano la maggioranza dei siti INFN;
  - formulazione di proposte implementative;
  - individuazione di strumenti di controllo/logging/prevenzione da integrare negli scenari proposti

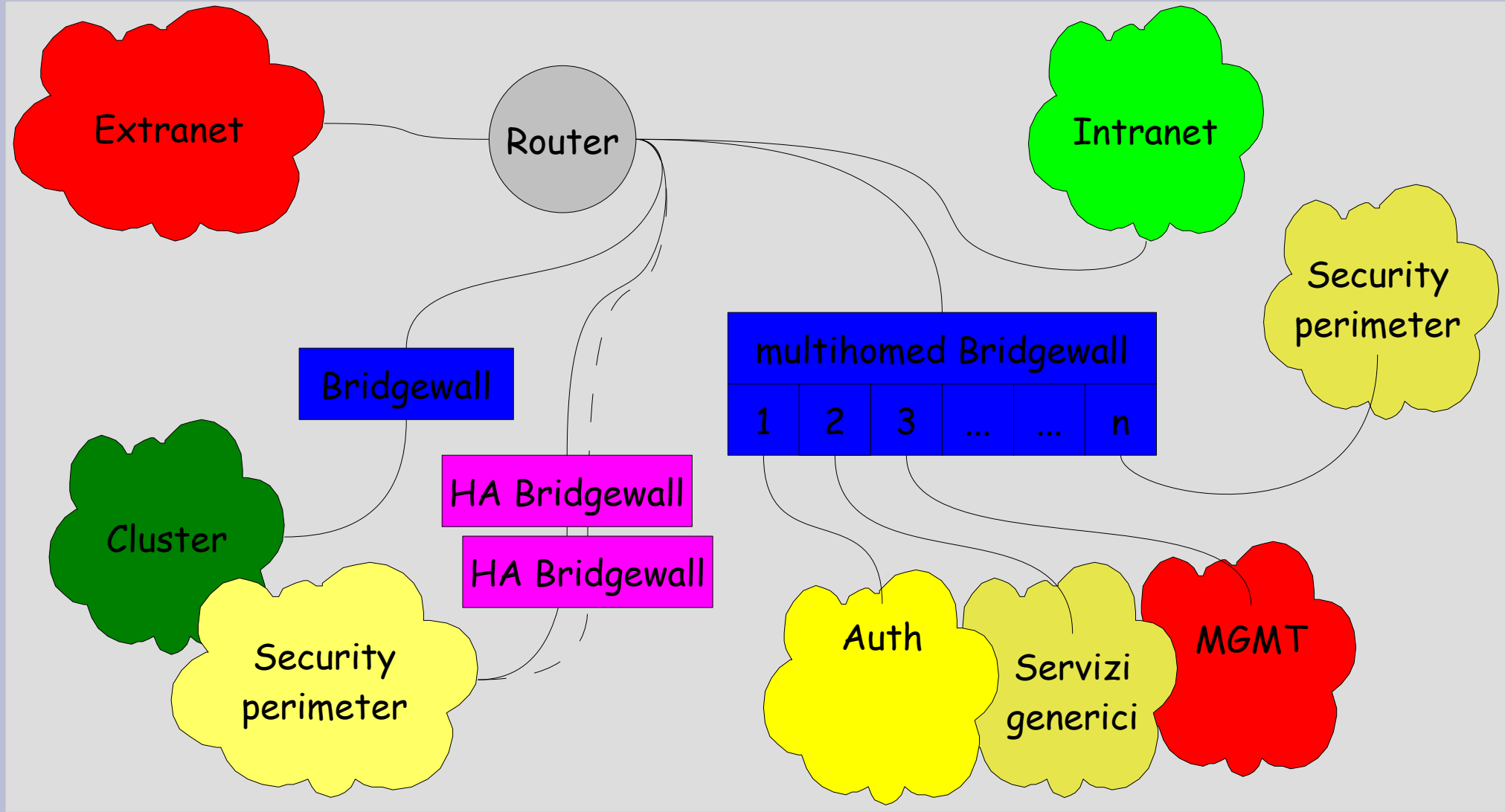
Il problema...

- E' possibile implementare una Security Policy (fisica & logica) tale da:
  - *rendere piu' sicura la rete senza sacrificare (troppo) prestazioni e comodita' d'accesso ai servizi, ed affrontare pesanti ristrutturazioni (ad es. routing)?*
  - non basare la sicurezza della rete su *buone pratiche* al di fuori del nostro controllo?
  - minimizzare la possibilita' di *effetto Domino?*
  - minimizzare il numero di punti di controllo/auditing?

... ed una sua possibile soluzione (forse l'unica)

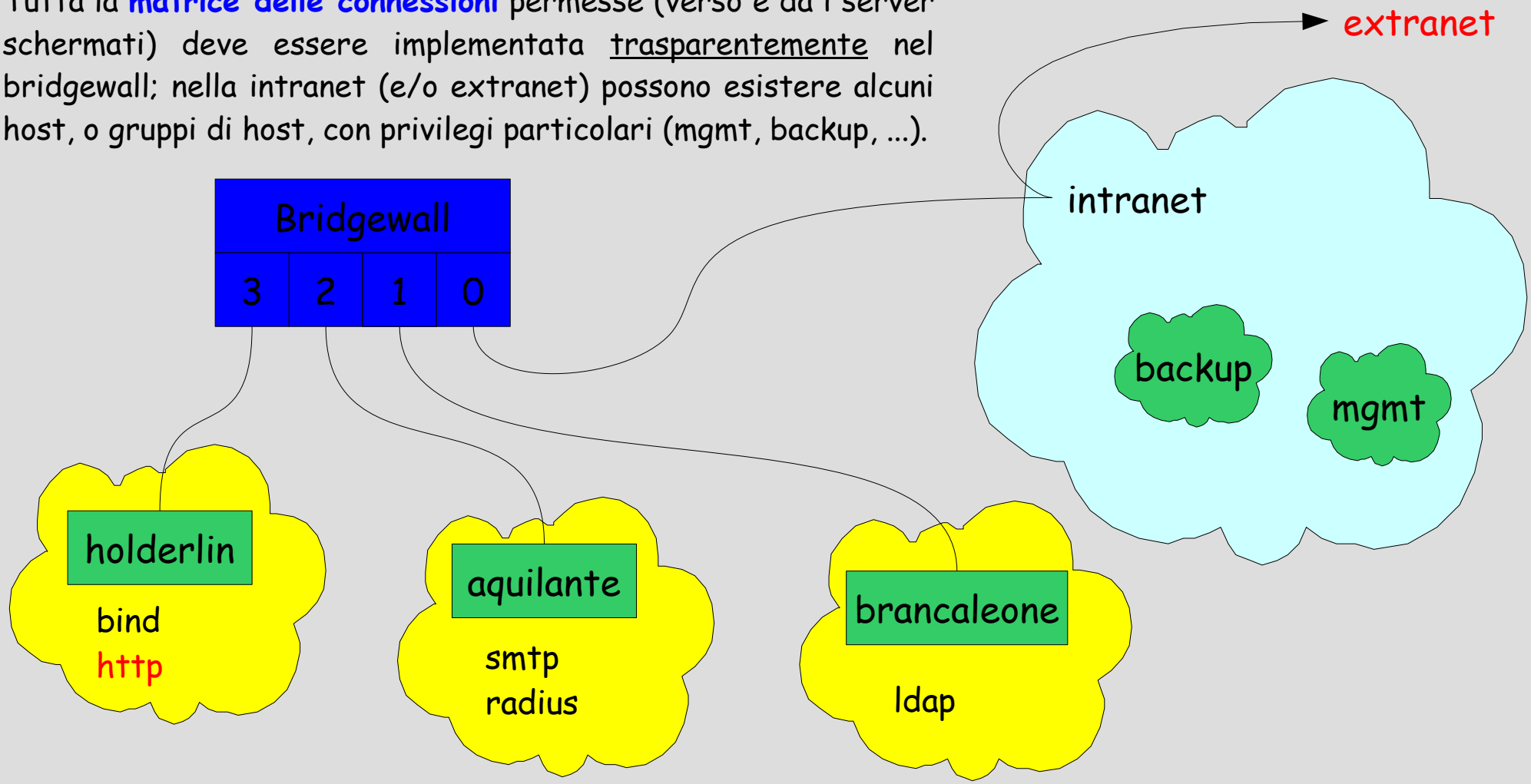
- **BRIDGEWALL = BRIDGE + FIREWALL**
  - stealth/transparent firewalling tramite apparato di livello 2 (**invisibile > inaccessibile**) con funzionalita' di inspection/filtering ai livelli 3,4,...,7 della pila OSI
  - implementabile con:
    - soluzioni Open Source (Linux bridge + ebtables/arptables/iptables/ipset/nf-hipac...)
    - soluzioni proprietarie (Cisco, Juniper, SonicWALL)

Topologia possibile

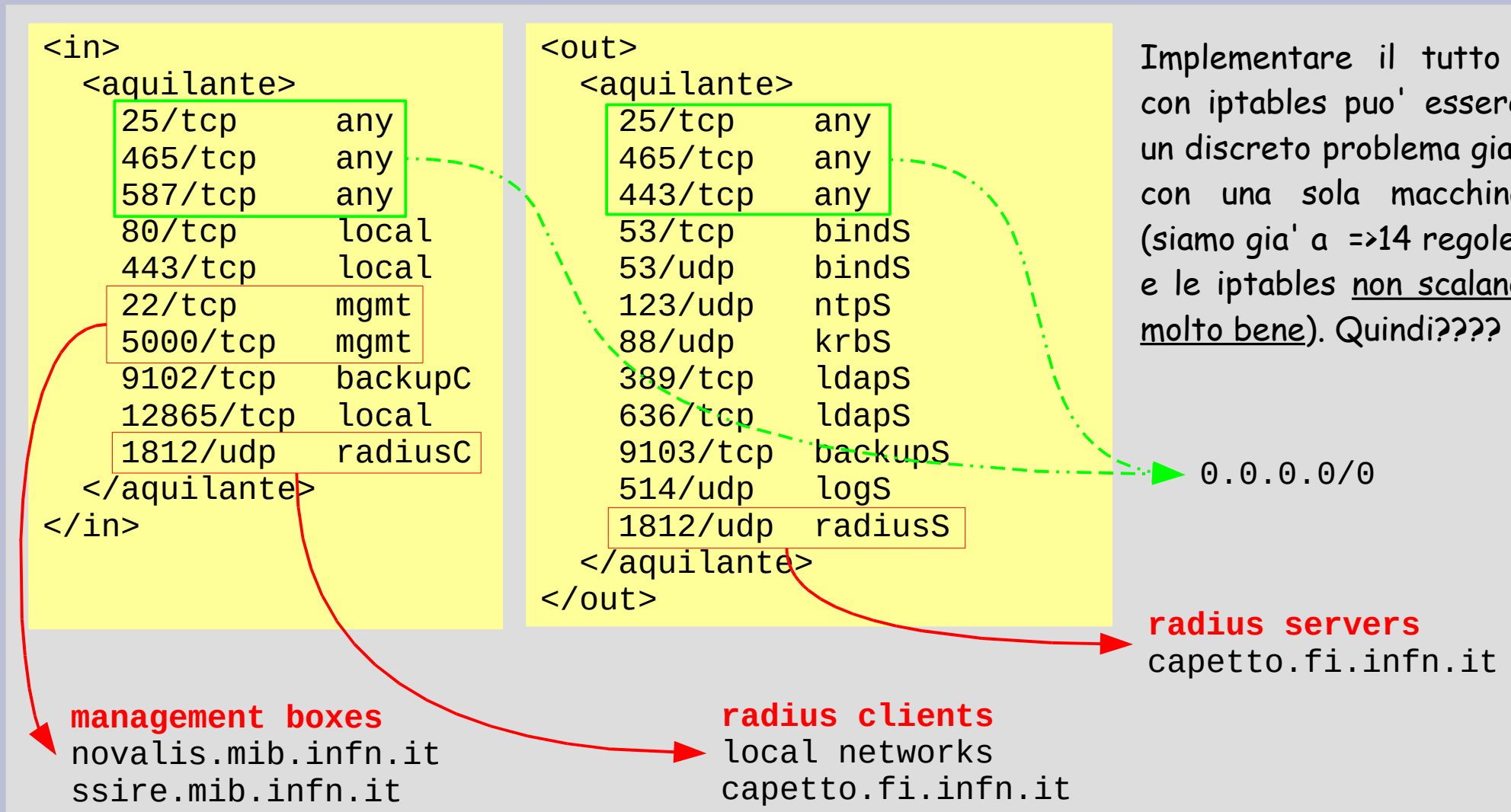


Linux BW testbed

Tutta la **matrice delle connessioni** permesse (verso e da i server schermati) deve essere implementata trasparentemente nel bridgewall; nella intranet (e/o extranet) possono esistere alcuni host, o gruppi di host, con privilegi particolari (mgmt, backup, ...).



Matrice delle connessioni



IPSET!!!

- Almost verbatim from http://ipset.netfilter.org/:

IP sets are a framework inside the Linux 2.4.x and 2.6.x kernel, which can be administered by the *ipset* utility. Depending on the type, currently an IP set may store IP addresses, (TCP/UDP) port numbers or IP addresses with MAC addresses in a way, which ensures lightning speed when matching an entry against a set.

If you want to:

•**store multiple IP addresses or port numbers and match against the collection by iptables at one swoop**;

•**dynamically update iptables rules against IP addresses or ports without performance penalty**;

•**express complex IP address and ports based rulesets with one single iptables rule and benefit from the speed of IP sets**

then ipset may be the proper tool for you.

A simple example about set & bindings

```
# targets: ipmap set
# ipmap type: memory range where each bit represents one IP address
ipset -N targets ipmap --network 193.206.156.0/23
ipset -A targets novalis.mib.infn.it
ipset -A targets ssire.mib.infn.it
ipset -A targets promiscuo.mib.infn.it
# ports: portmap set
# portmap type: memory range where each bit represents one port
ipset -N ports portmap --from 1 --to 1024
ipset -A ports 22
ipset -A ports 25
ipset -A ports 80
# bind 'ports' set to novalis, ssire
ipset -B targets ssire -b ports
ipset -B targets novalis -b ports
# iptables rule using the set match & bindings
...
iptables -A FORWARD -m set --set servers dst,dst -j ACCEPT
...
```

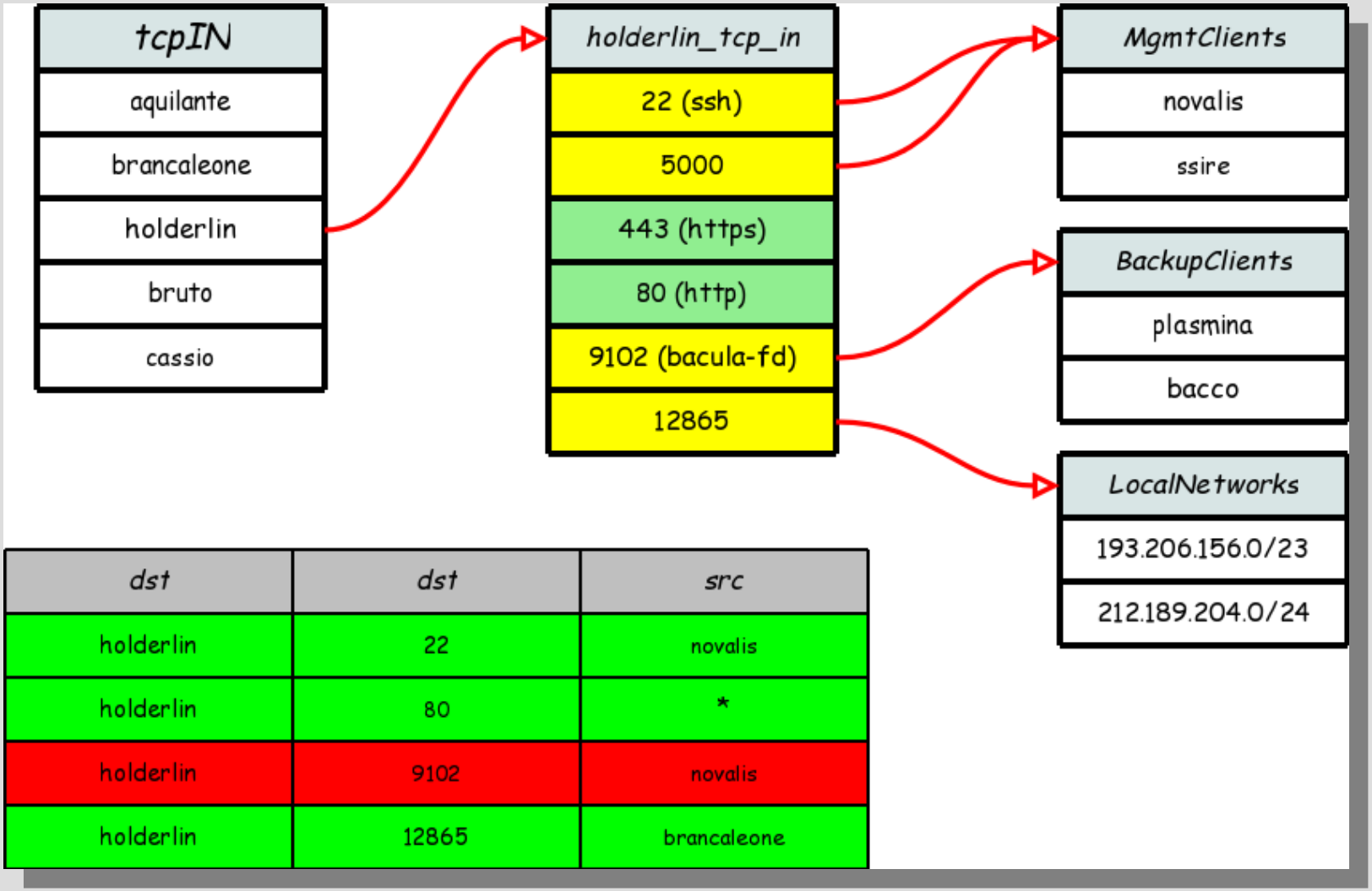
Firewall will forward pkts destined to any port on *promiscuo*, while only ports 22,25,80 will be reachable on *ssire* & *novalis* >1 line, N matches.

Bridgewall rules w/ipset

```
Chain PREROUTING (policy ACCEPT) # raw table
target     prot src dst
NOTRACK    all  *   *   PHYSDEV match   physdev-in UPLINK ! Set ScreenedHosts dst
DROP       all  *   *   PHYSDEV match   physdev-in UPLINK ! Set ScreenedHosts dst
Chain FORWARD (policy DROP) # filter table
target     prot src dst
ACCEPT     all  *   * state RELATED,ESTABLISHED
DROP       all  *   * state INVALID
LOG_DENY   all  *   * state NEW set hostDeny src
LOG_DENY   all  *   * state NEW set portDeny dst
TCPSRC     tcp  *   * state NEW PHYSDEV match ! --physdev-in UPLINK
TCPDST     tcp  *   * state NEW PHYSDEV match ! --physdev-out UPLINK ...
UDPSRC     udp  *   * state NEW PHYSDEV match ! --physdev-in UPLINK
UDPDST     udp  *   * state NEW PHYSDEV match ! --physdev-out UPLINK ...
ACCEPT     icmp *   * PHYSDEV match ! --physdev-in UPLINK icmp echo-request
LOG_DROP   all  *   *
Chain TCPDST # filter table
target     prot src dst
ACCEPT     all  *   *   set tcpIN dst,dst,src
LOG_DROP   all  *   *
Chain LOG_DROP # filter table
target     prot src dst
SET        all  *   *   ! set hostDeny src add-set hostDeny src
LOG        all  *   *   ...
DROP       all  *   *   ...
```

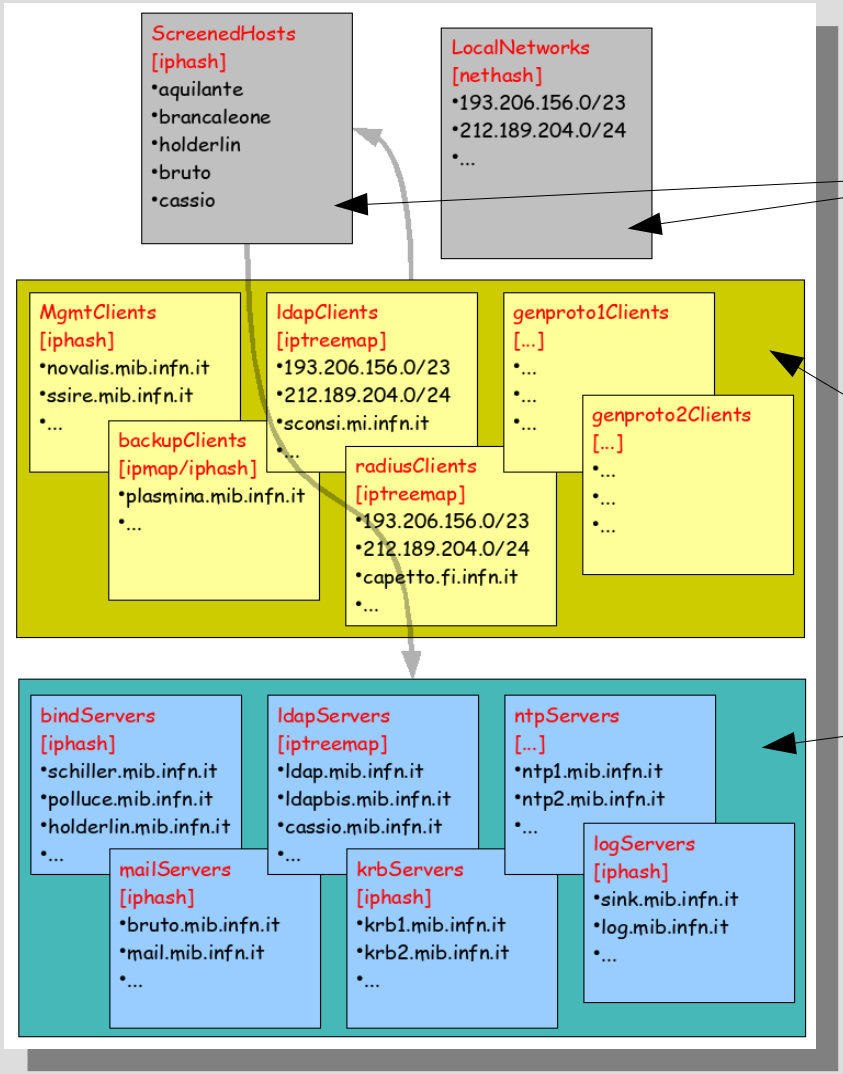
Dynamic update: *bad hosts* added to hostDeny set at runtime

... set tcpIN dst,dst,src ...



dst	dst	src
holderlin	22	novalis
holderlin	80	*
holderlin	9102	novalis
holderlin	12865	brancaleone

Sets, sets, sets, ...



Basic sets:

- ScreenedHosts
- LocalNetworks

Clients (in)

Servers (out)

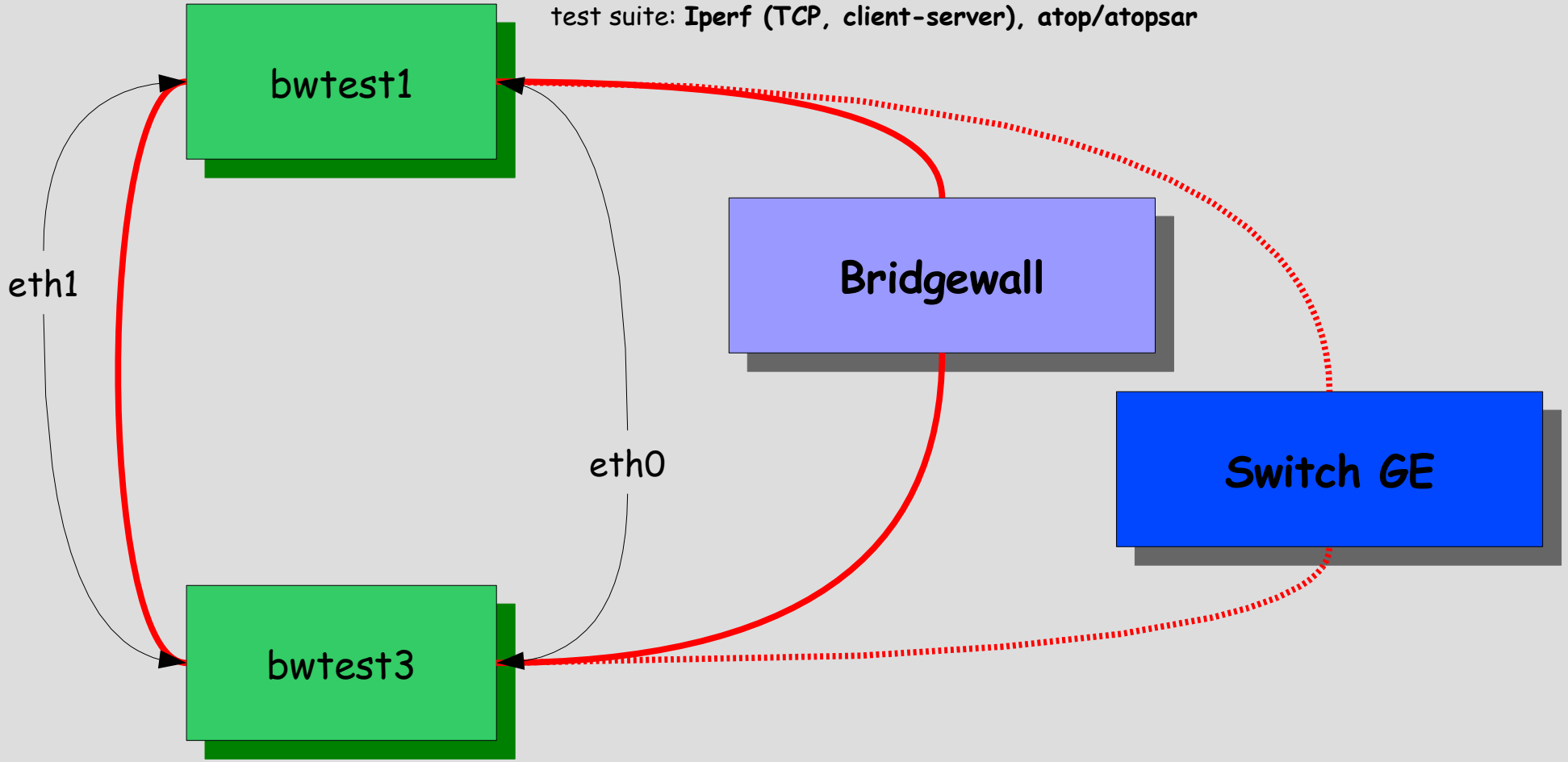
on test BW: 34 sets (max 256, configurable parameter)

Bridgewall: hardware & software

- XEON dual core 3060 @ 2.40GHz
- 2GB RAM
- 4xIntel Gbit Eth NIC (1 irq line/NIC, irq -> CPU#0)
  - 2x82573 on board
  - 2x82571 on PCI-E 4x dual port card
- Fedora 7, kernel 2.6.23 (~ FC-out-of-the-box + *ipset*)
- e1000 driver: 7.3.20-k2-NAPI
- iptables 1.3.8, ipset 2.3.0, ebtables 2.0.8
- netfilter tweaking:
  - **option nf_conntrack: hashsize=262144**
    - **nf_conntrack_buckets = 262144**
    - **nf_conntrack_max = 262144**

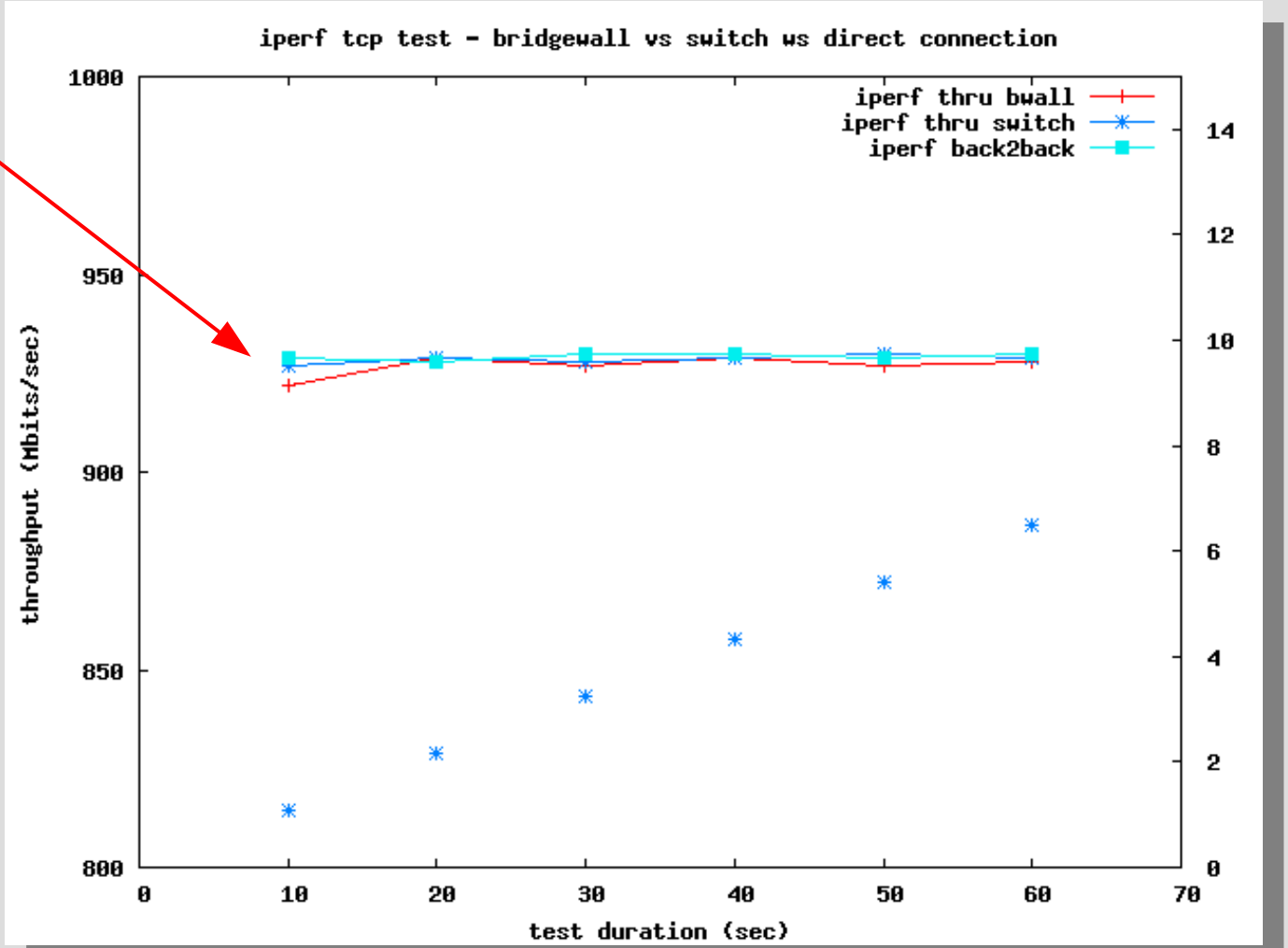
Throughput - test setup

bwtest1, bwtest3, Bridgewall: Xeon 3060 @ 2.40GHz, 2GB RAM, Intel 825XX PCI-E
Switch GE: Nortel 5510-48T - 48 x 10/100/1000
test suite: **Iperf (TCP, client-server), atop/atopsar**

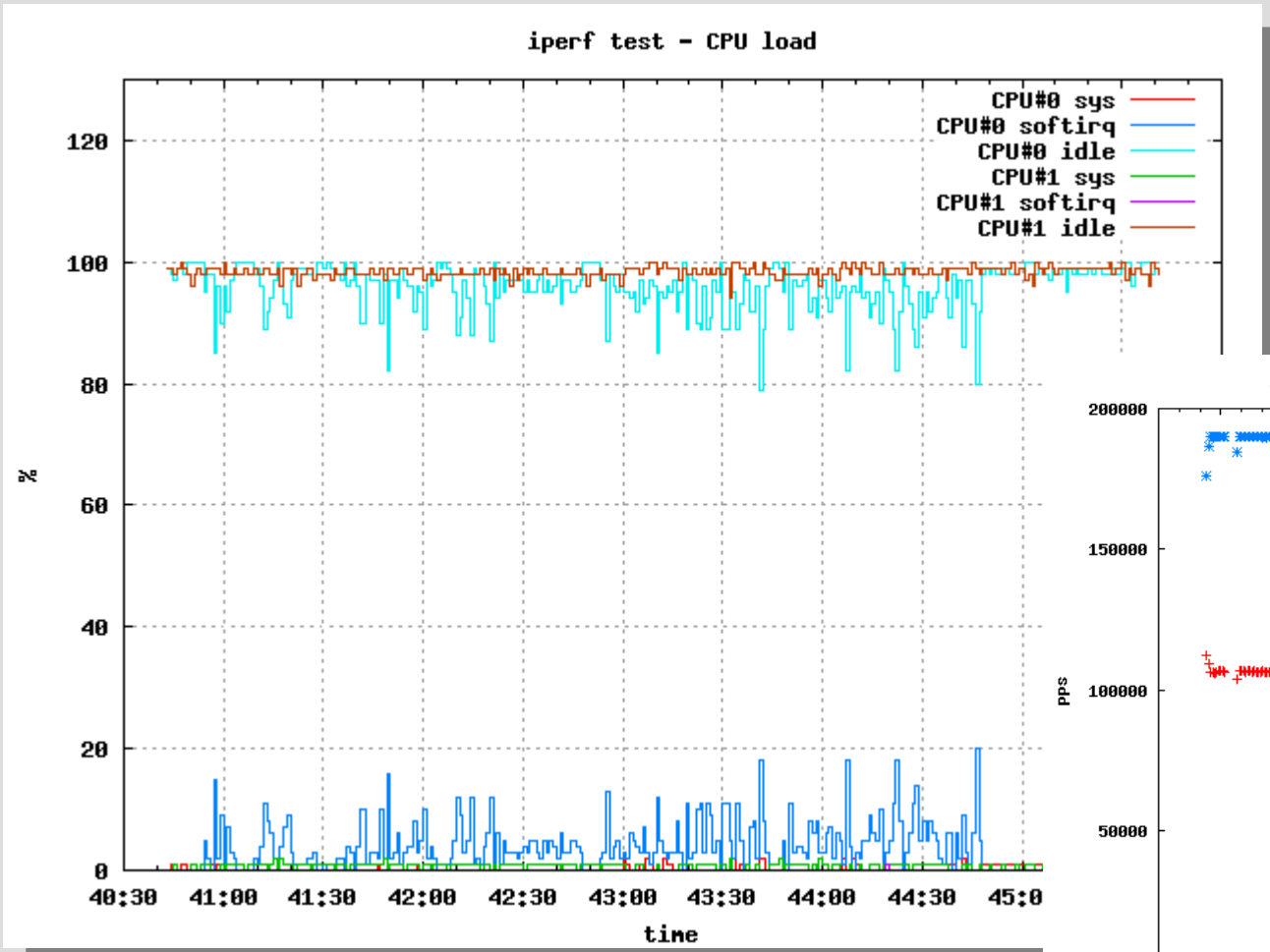


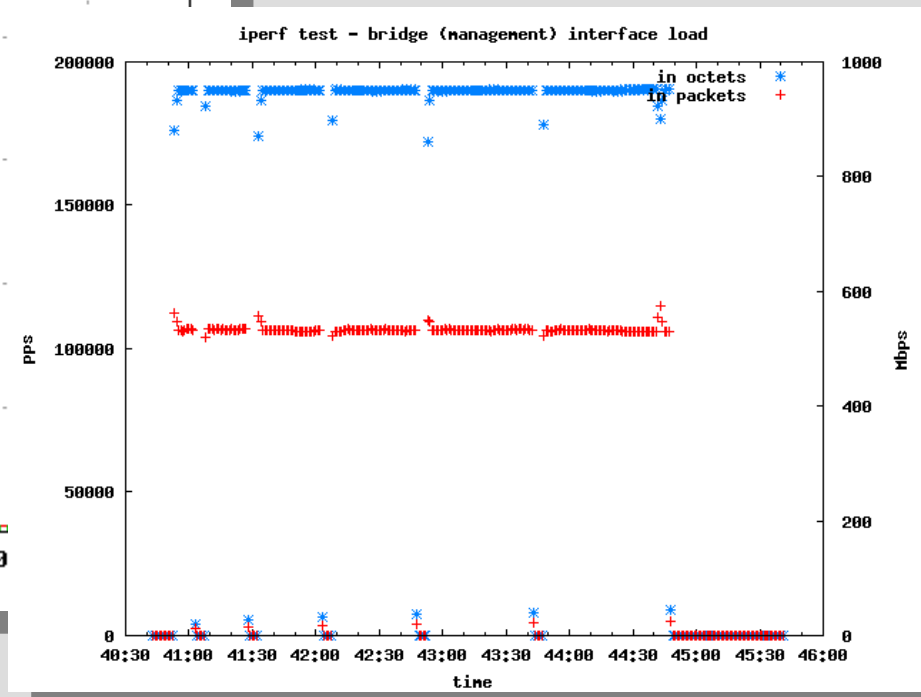
Throughput - BW vs switch vs b2b

~930 Mbps



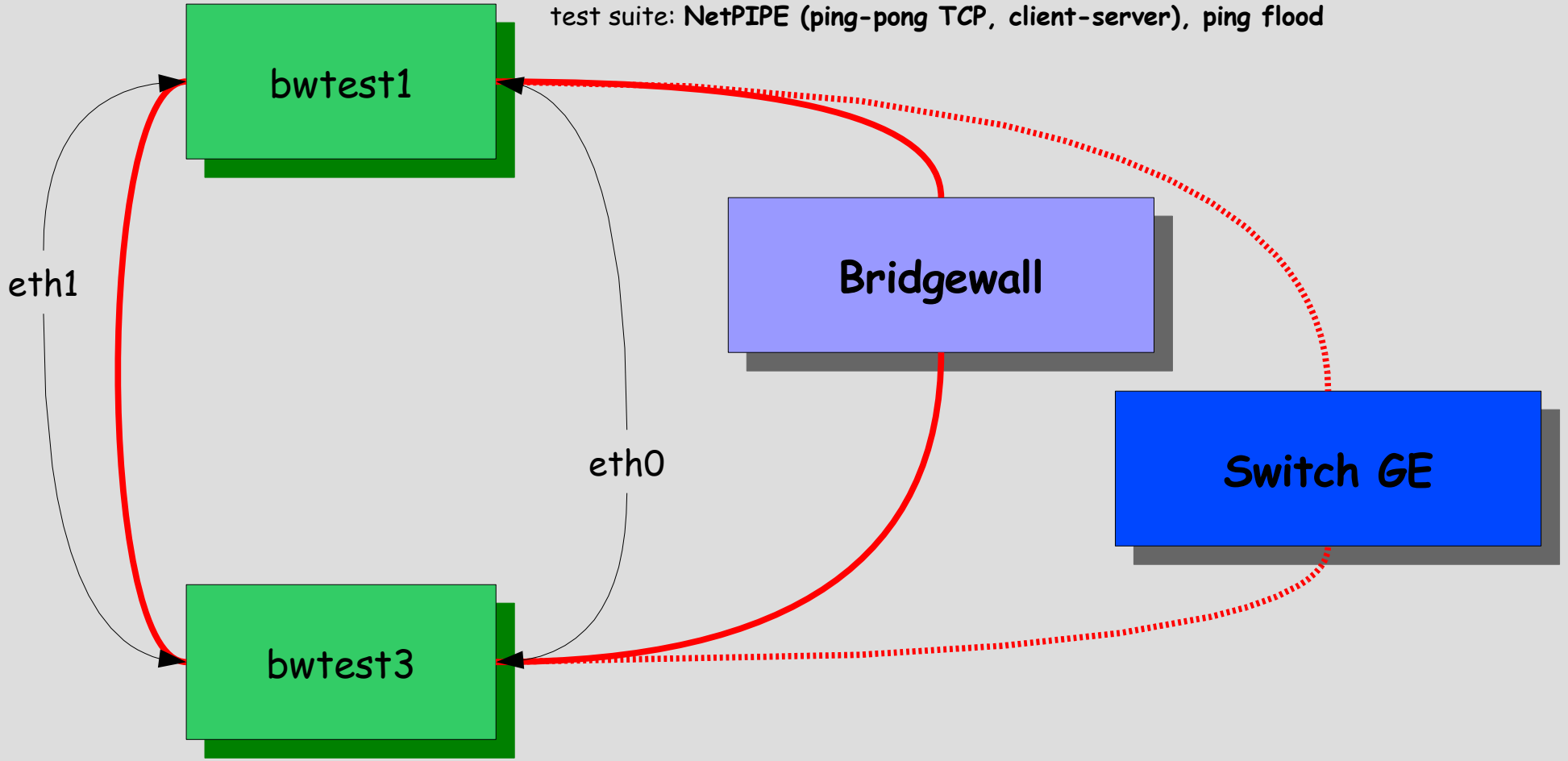
Throughput - BW CPU & net load



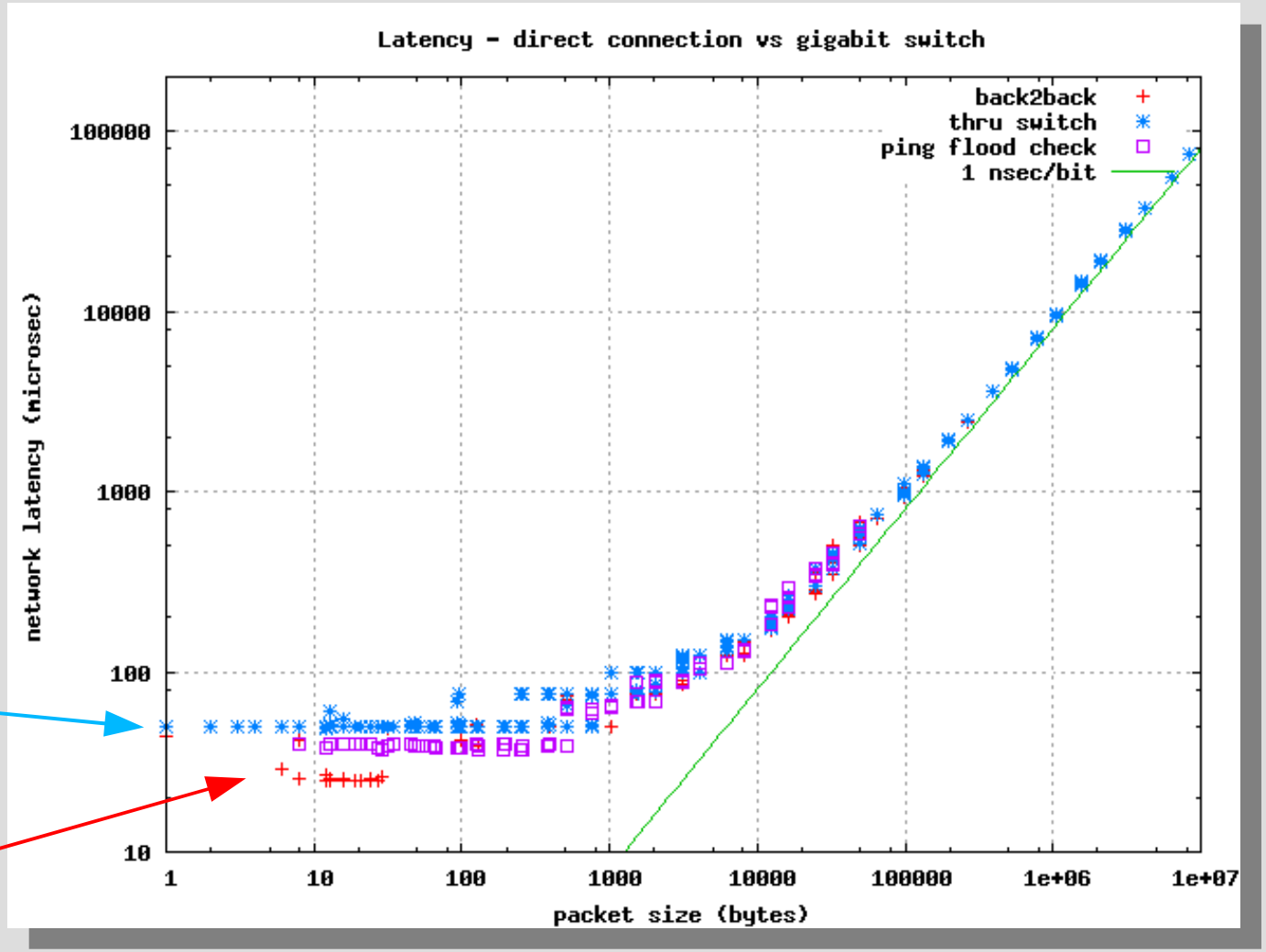


Latency - test setup

bwtest1, bwtest3, Bridgewall: Xeon 3060 @ 2.40GHz, 2GB RAM, Intel 825XX PCI-E
Switch GE: Nortel 5510-48T - 48 x 10/100/1000
test suite: **NetPIPE (ping-pong TCP, client-server), ping flood**



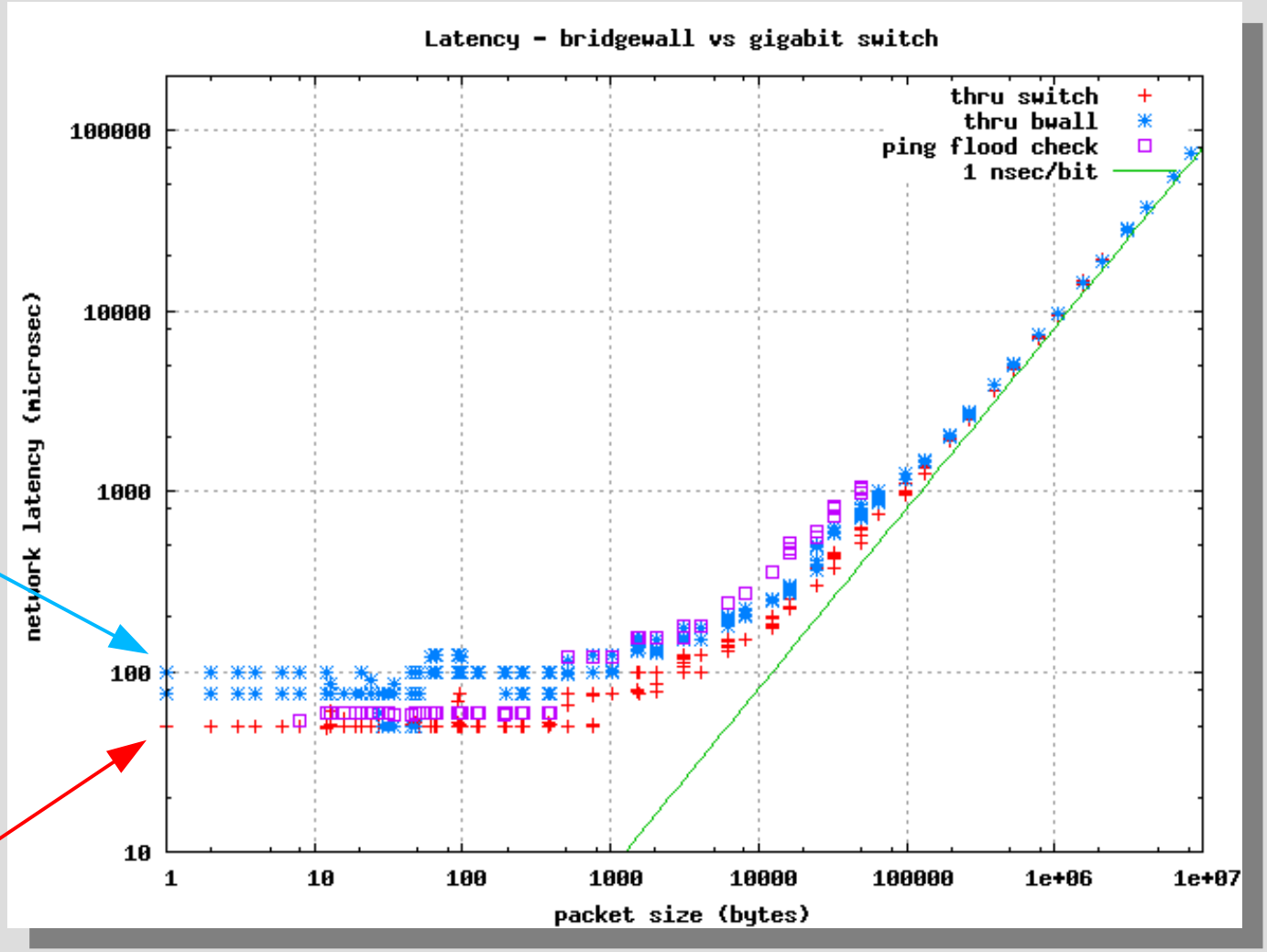
Latency - back2back vs switch



~50 µs

~25-50 µs

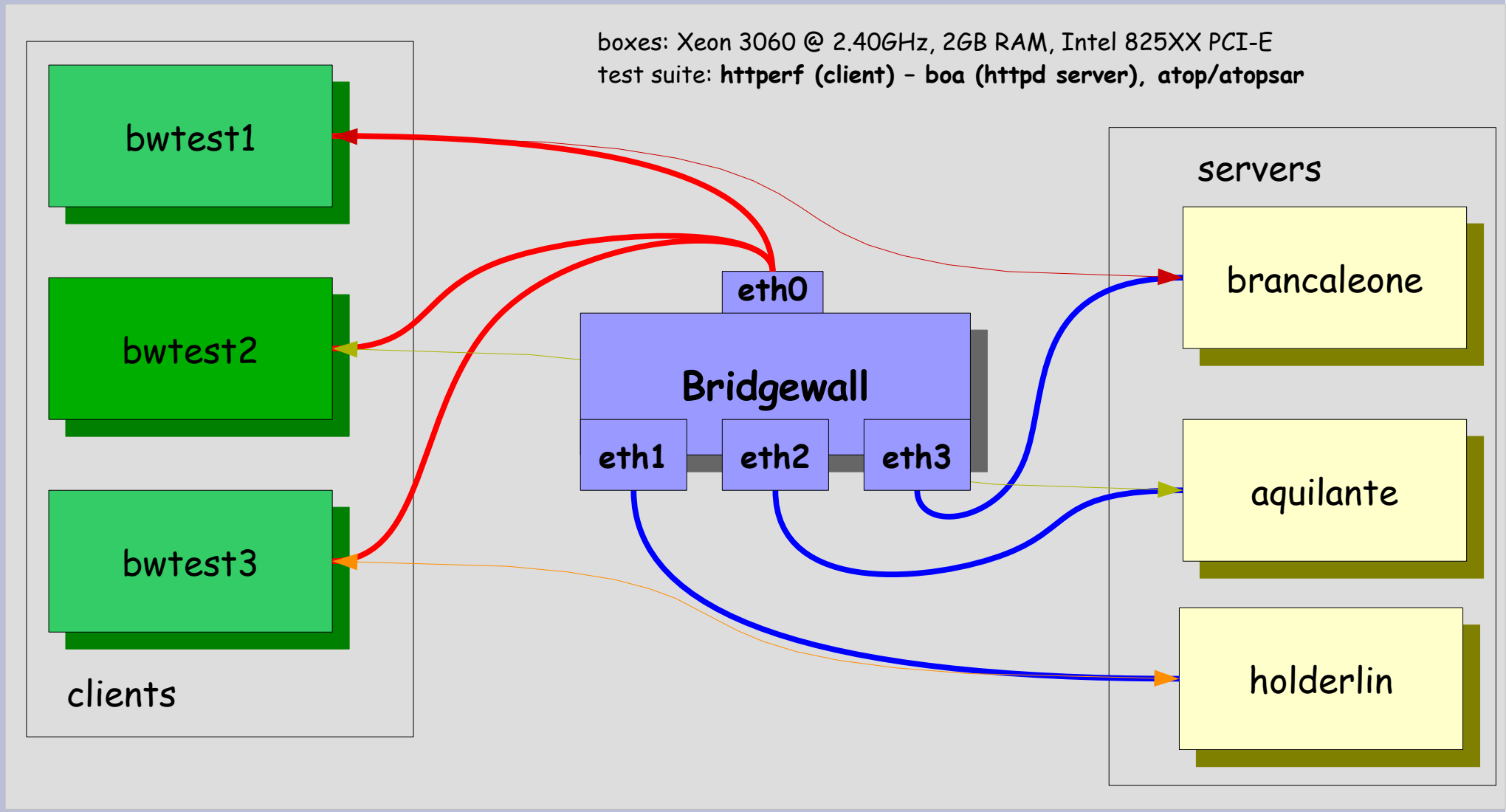
Latency - bridgewall vs switch



~75-100 µs

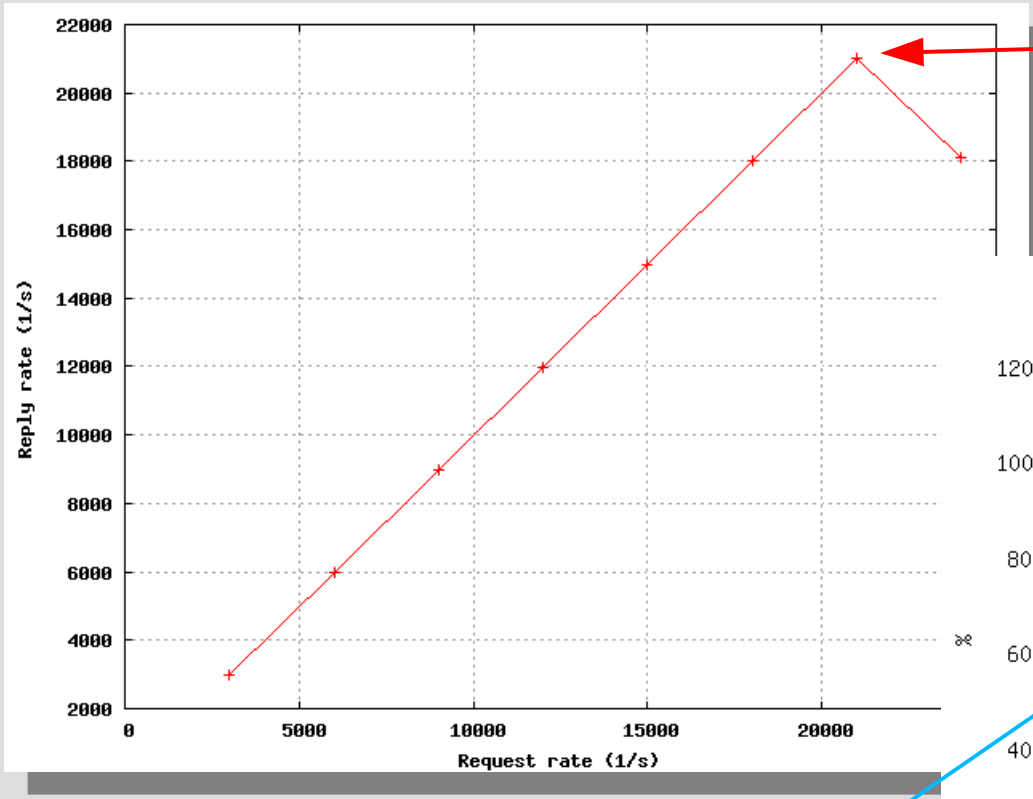
~50 µs

Connection rate - test setup

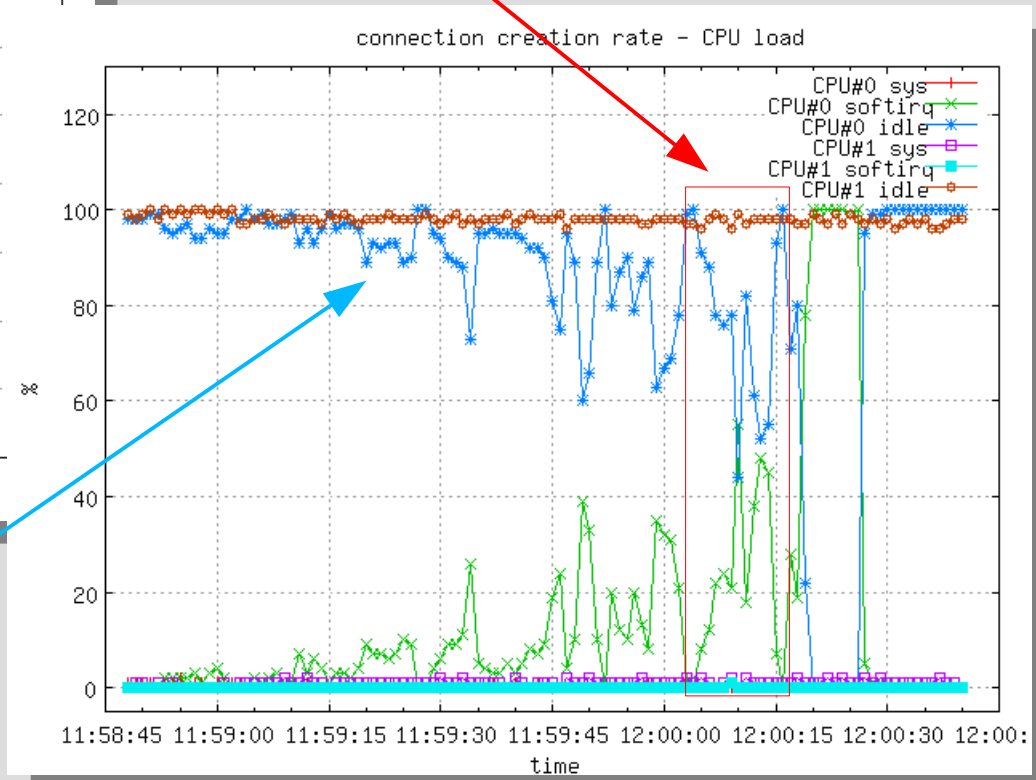


Connection rate - avg performance

[6 runs 1k->8k req/sec/client]



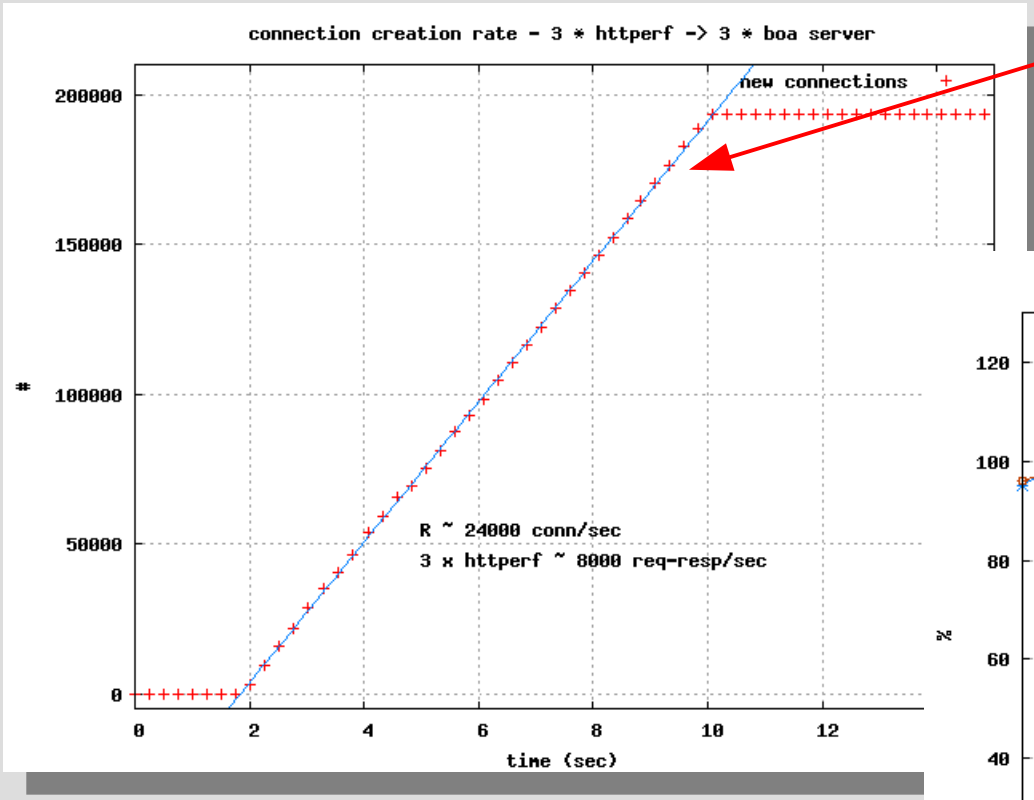
~21000 (new) cps @ CPU ~50%



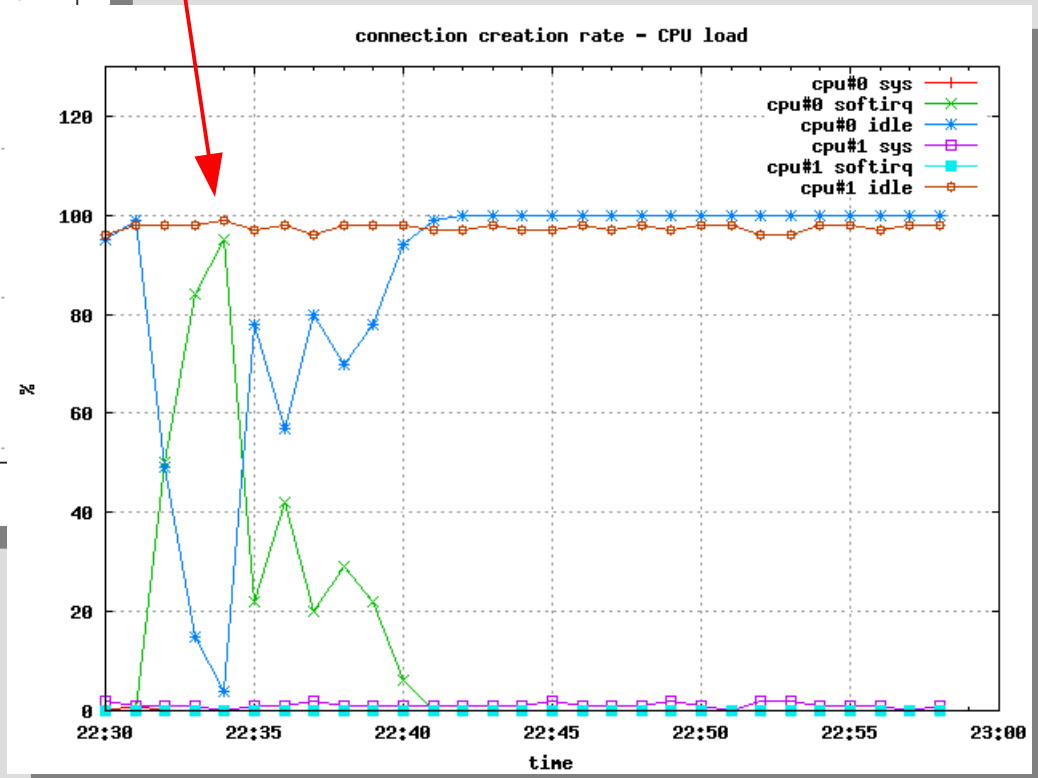
CPU < ~10% up to ~ 10000 (new) cps

Connection rate – peak performance

[single shot - 8k req/sec/client]



peak: ~24000 (new) cps @ CPU ~ 100%



"Netfilter Performance Testing"

Kadlecsik & Pasztor

- H/W: Dual Opteron @ 2.2GHz, 4GB RAM, Intel 82546EB Gbit Ethernet (dual), kernel 2.6.11 with NAPI
- Testbed: http request-reply test, 160 clients (httperf) / 160 servers (boa) [=> 3500000 max concurrent connections.]
- iptables/ipset/nf-hipac comparison: iptables doesn't scale (the more the rules, the worse the performace); *ipset & nf-hipac perfom almost indifferently with regard of the number of rules (ipset is a tiny bit better than nf-hipac).*

	ROUTING	CONNTRACK	CONNTRACK/NF
cps	55000	25000	25000
pps	700000	330000-340000	300000

Performances: proprietary solutions vs LBW

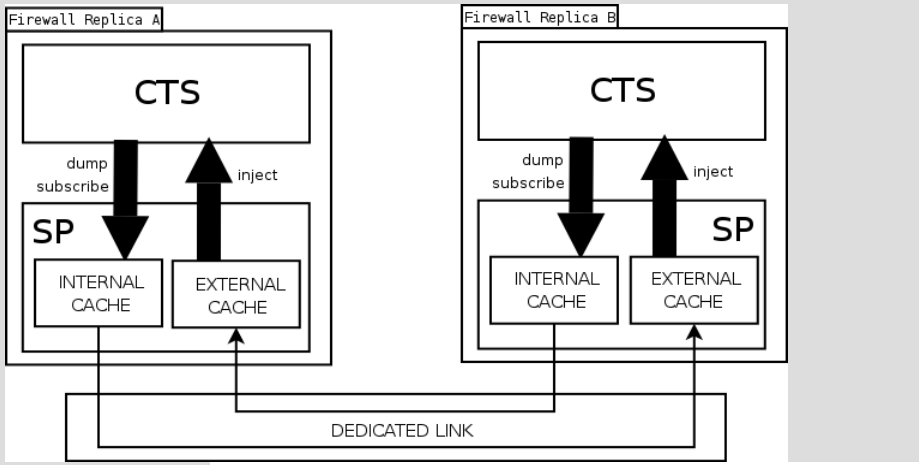
	Cisco ASA 5520	Juniper NetScreen 208	SonicWALL PRO 4060	LBW
Max firewall throughput (Mbps)	450	375	300	930
Latency (microSec)	?	?	?	75 - 100
Max connections	280000	128000	524288	>262144
Max connections/second	9000	11500	5000	21000
Integrated ports	4x10/100/1000+1x10/100	8x10/100	6x10/100	*
Expansion slot	yes (4GE, CSC, AIP)	no	no	*
L2 transparent firewalling	yes	yes	~yes	yes
Sec. contexts/zones - VFW	2 - 20 (licensed feature)	8 - 18 (upgrade)	yes (PRO1260: 24!!)	-
Stateful HA support	Act/Act & Act/Stb	Act/Act & Act/Stb	Act/Stb	Act/Stb
Price	10.1k (4GE: ~6.3k)	11.5k	???	1.5k

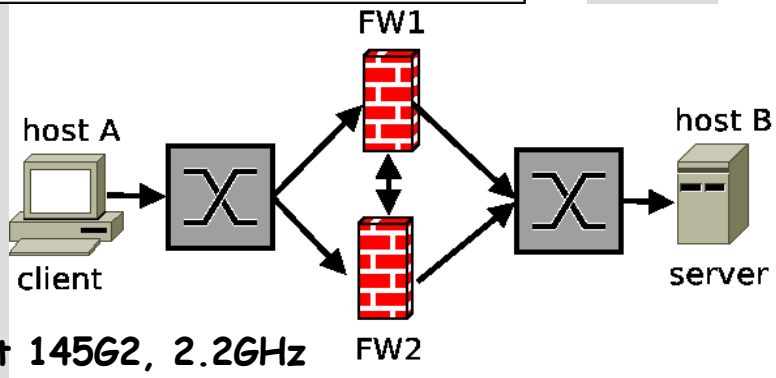
CSC: Content Security & Control
AIP: Advanced Inspection & Prevention

bridge + conntrack + netfilter

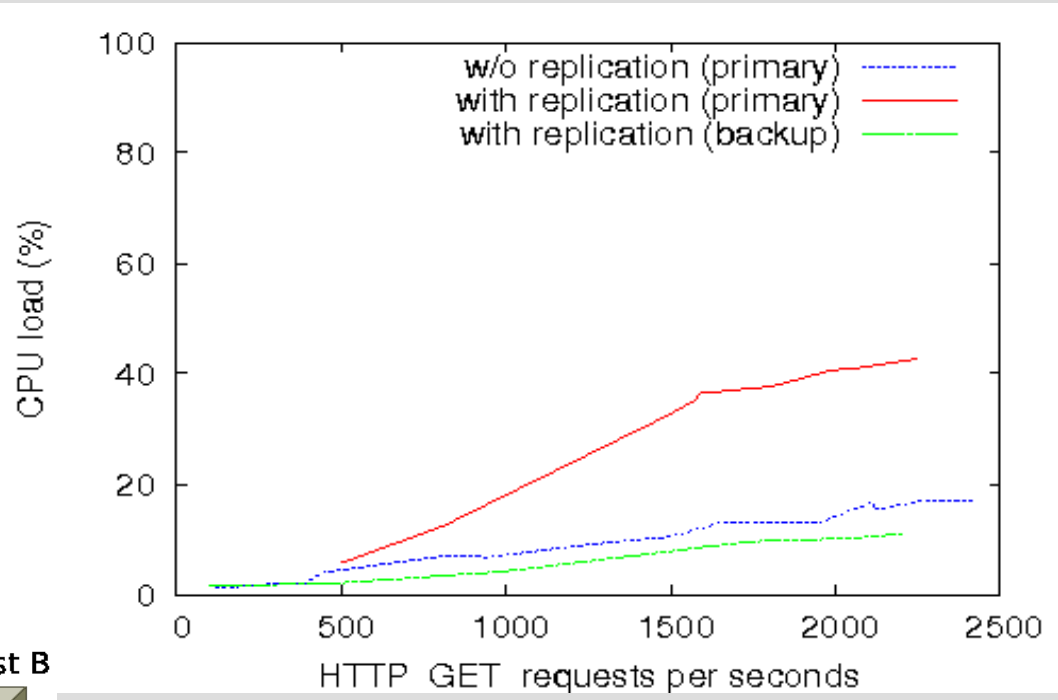
HA Bridgewall: conntrackd

- From *"Conntrack-tools: HA for stateful Linux firewalls"*, by Pablo Neira





HP Proliant 145G2, 2.2GHz



Auth. failover <u>without established connections loss</u>

- Primary/Backup, Multiprimary
- protocols: alarm-based, reliable UDP

Spunti, cose da fare
(se c'e' interesse)

- script per creazione zone/regole (c'e': *behemot...*);
- ottimizzazione kernel/SO/regole (diskless?)
- Bridgewall virtuali?
- Dual core -> Quad core (IRQ distribuiti???)
- Opteron (per-cpu memory controller)
- Intel quad-port adapter (4+4+2 ports bridgewall?)
- Policy dinamiche: integrazione IDS - BW w/dynamic ipsets; rilevamento/blocco *anche* attacchi dall'interno
- nf-HIPAC
- conntrackd (dual port HA bridgewall?)
- Test Cisco ASA? SonicWALL entry-level?

Conclusioni (parziali)

- I bridgewall (BW) possono integrarsi facilmente e trasparentemente (e senza richiedere alcuna operazione di riconfigurazione degli schemi di naming/routing) in strutture di qualsiasi complessita' per isolare/proteggere con la granularita' desiderata servizi fondamentali o gruppi di macchine, o definire in una LAN perimetri di sicurezza multipli ed indipendenti (cluster etc.).
- Le soluzioni open-source garantiscono buone prestazioni, stabilita', flessibilita', ridondanza a costi contenuti.
- Soluzioni proprietarie interessanti, ma da provare.