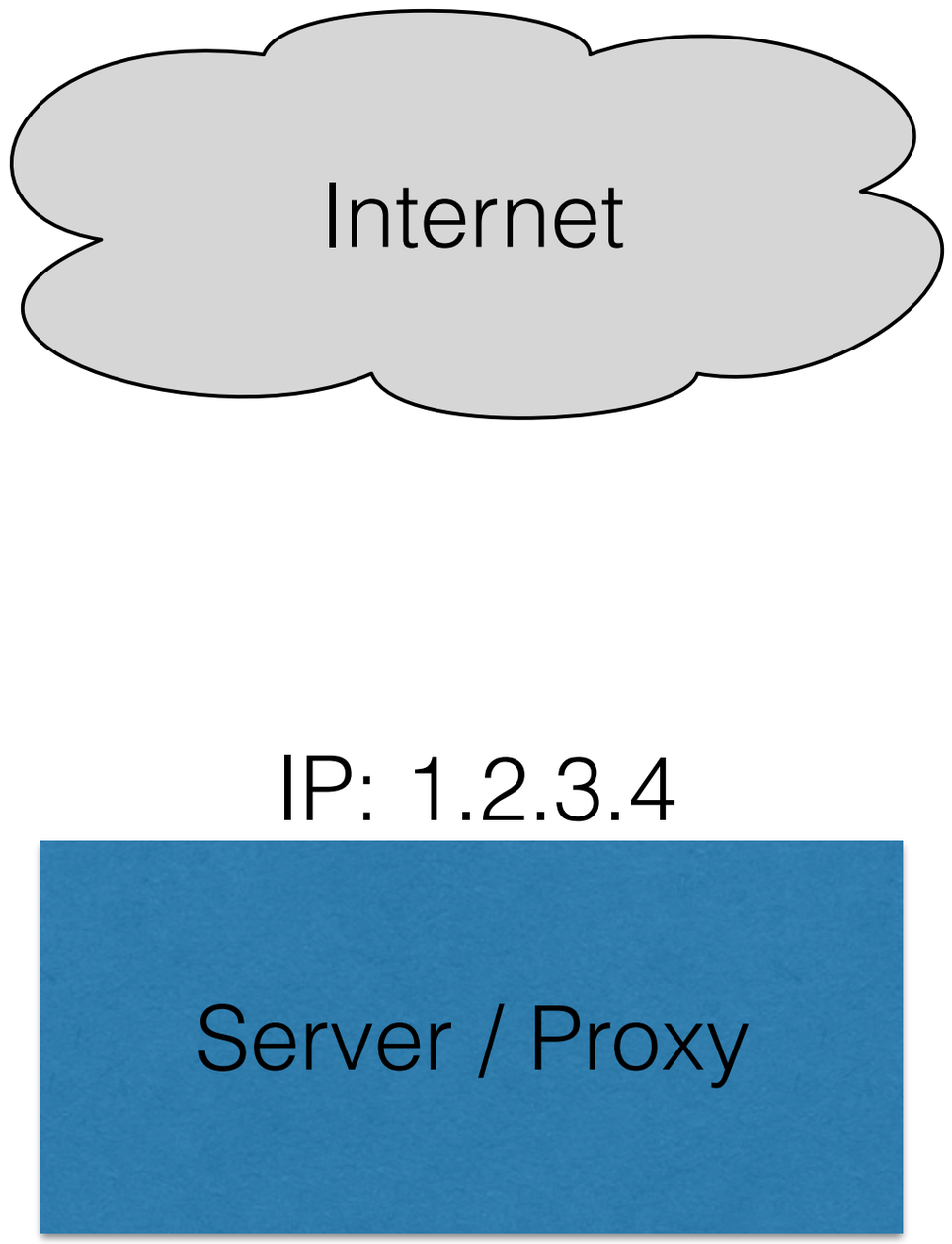
Multipath TCP behind Layer-4 loadbalancers

draft-paasch-mptcp-loadbalancer

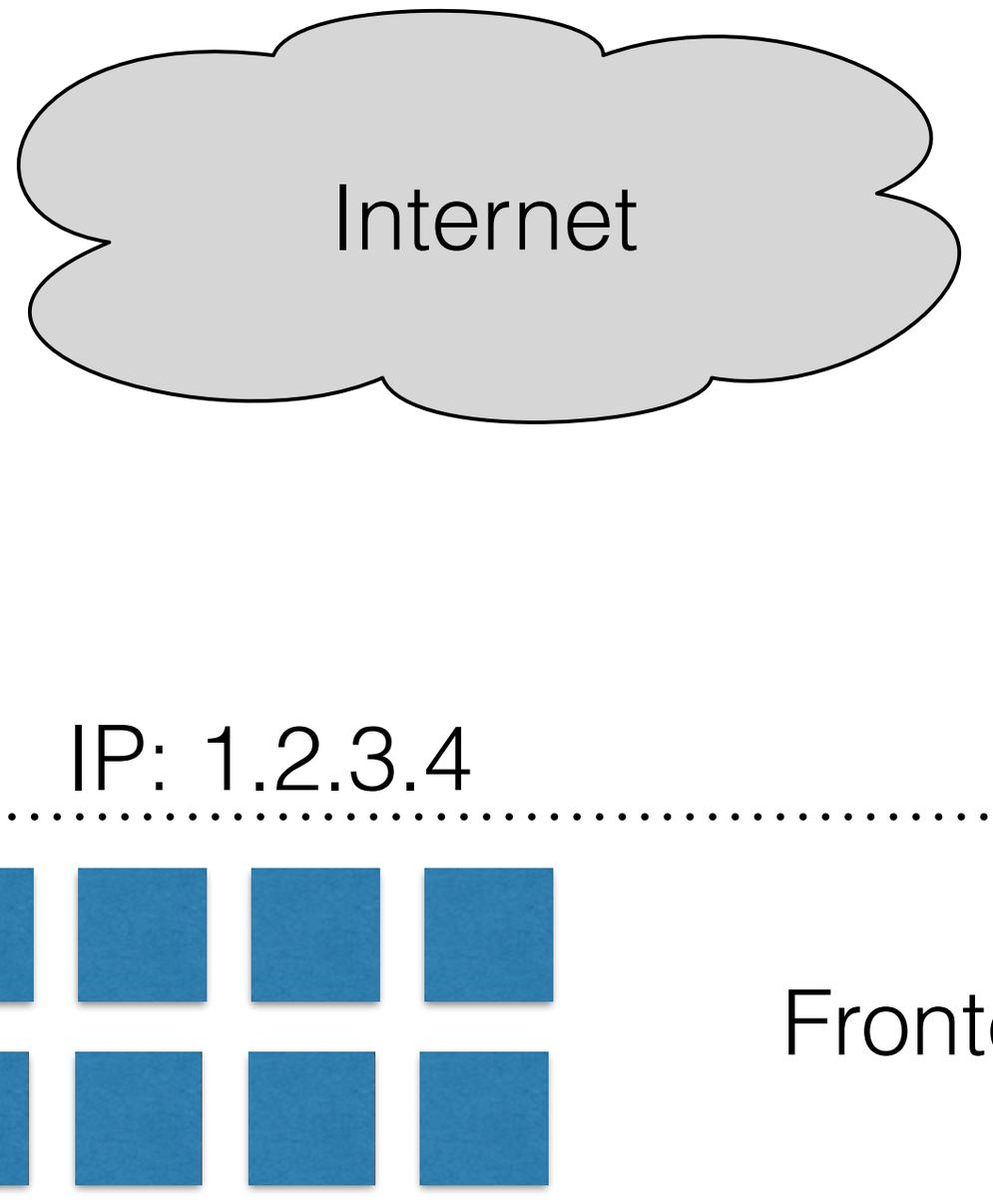
Christoph Paasch <cpaasch@apple.com>
Greg Greenway <ggreenway@apple.com>
Alan Ford <alan.ford@gmail.com>

Loadbalancer infrastructure

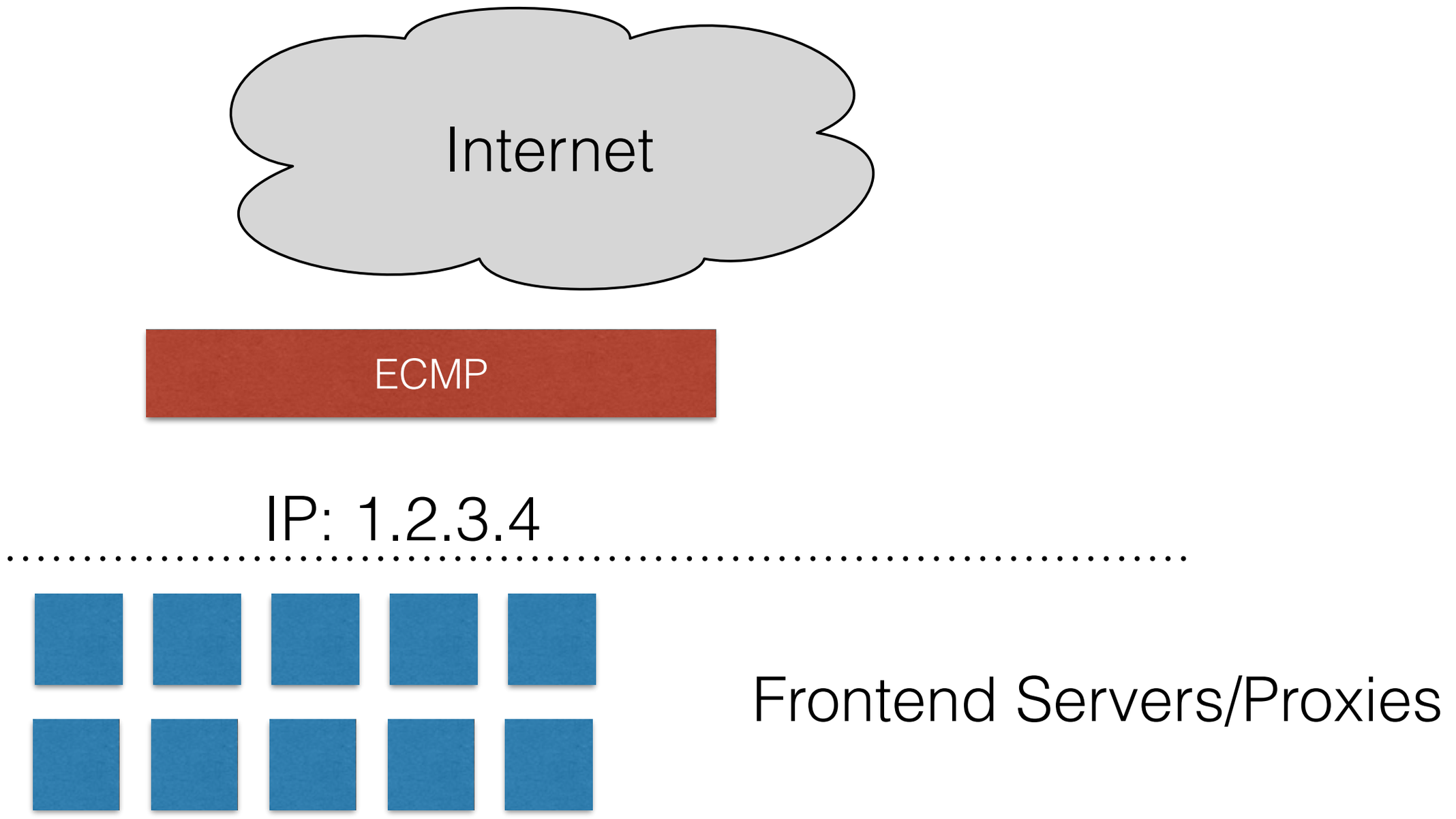
Classical loadbalancer infrastructure



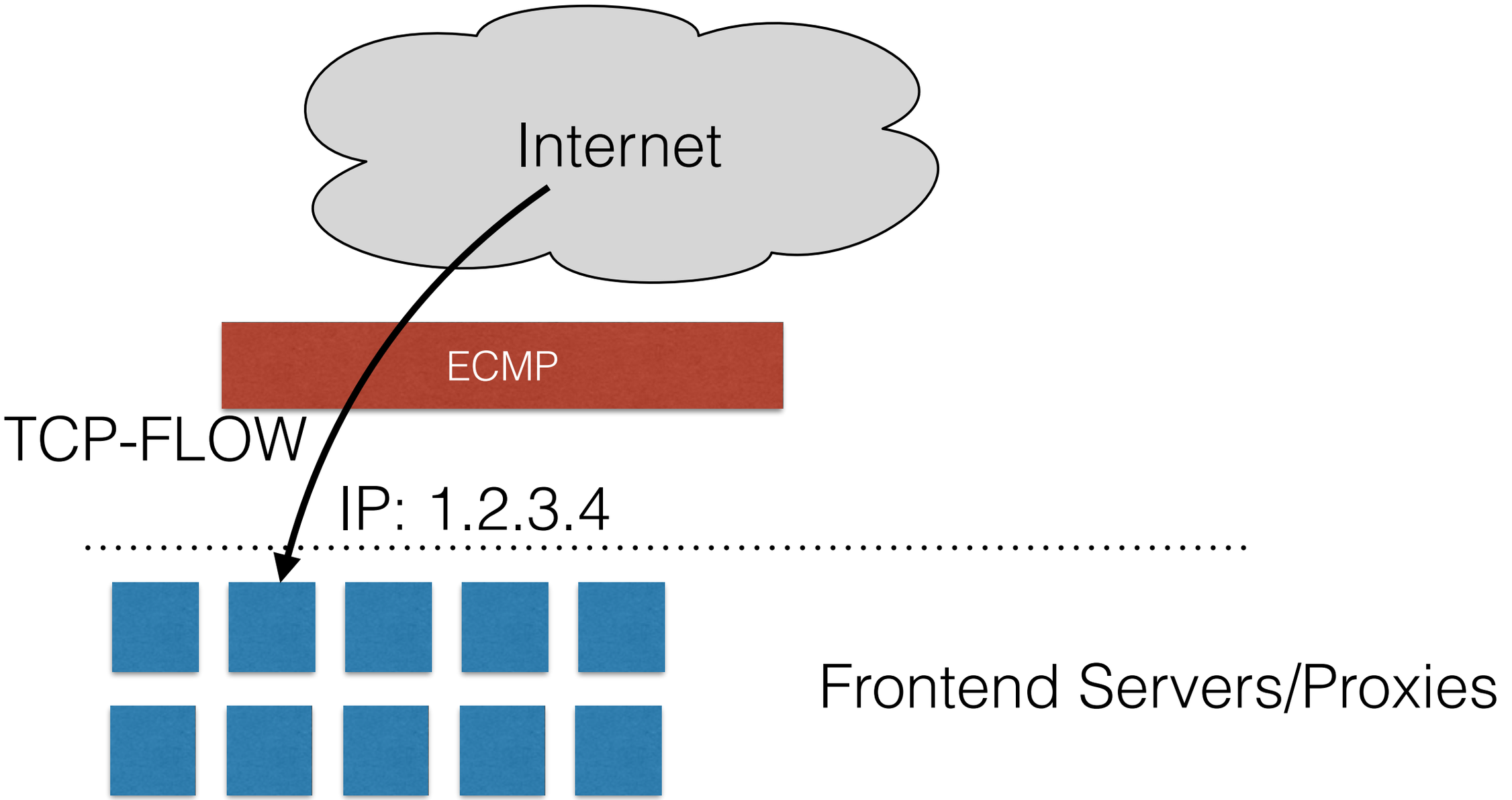
Classical loadbalancer infrastructure



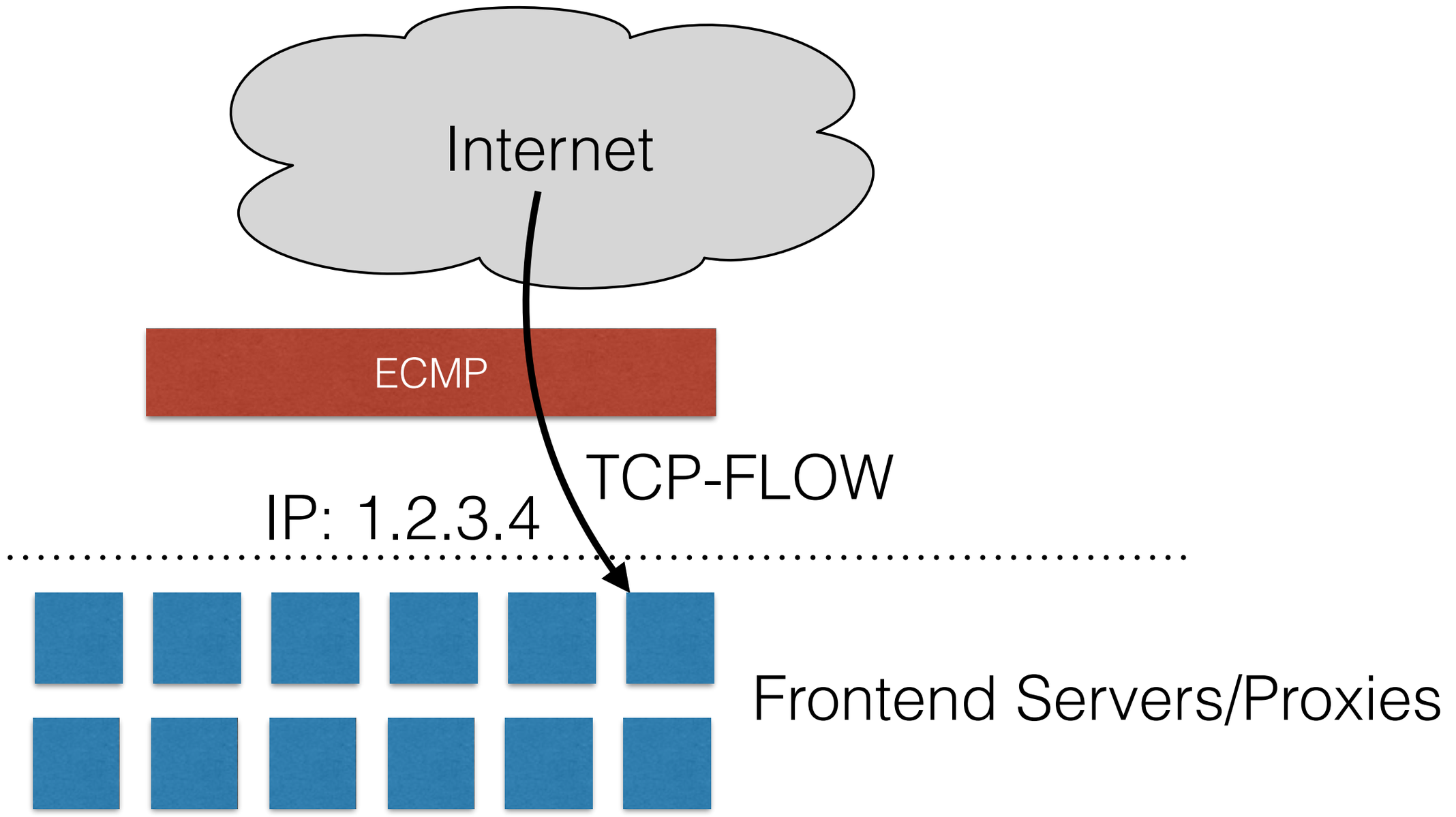
Classical loadbalancer infrastructure



Classical loadbalancer infrastructure

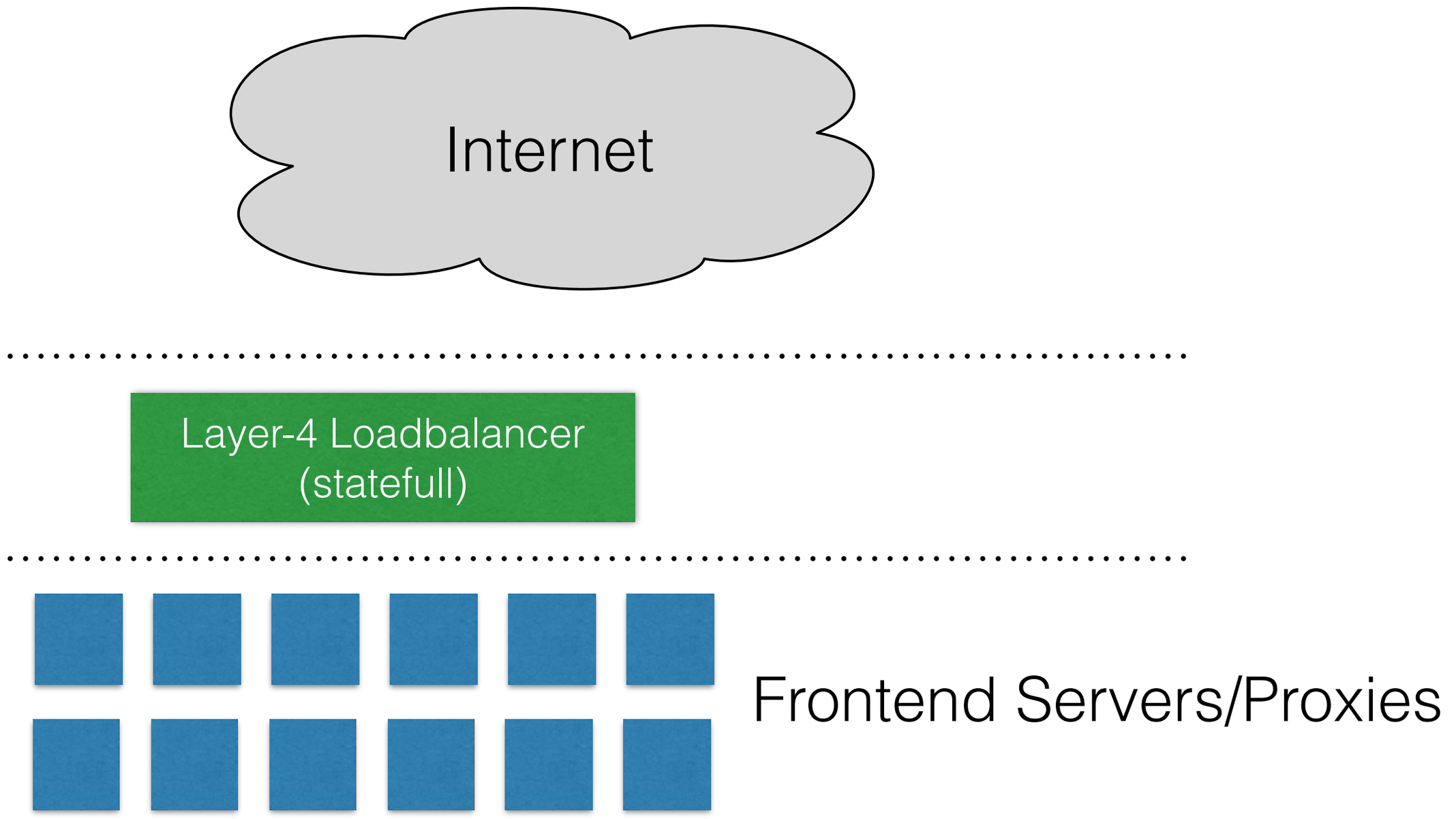


Classical loadbalancer infrastructure



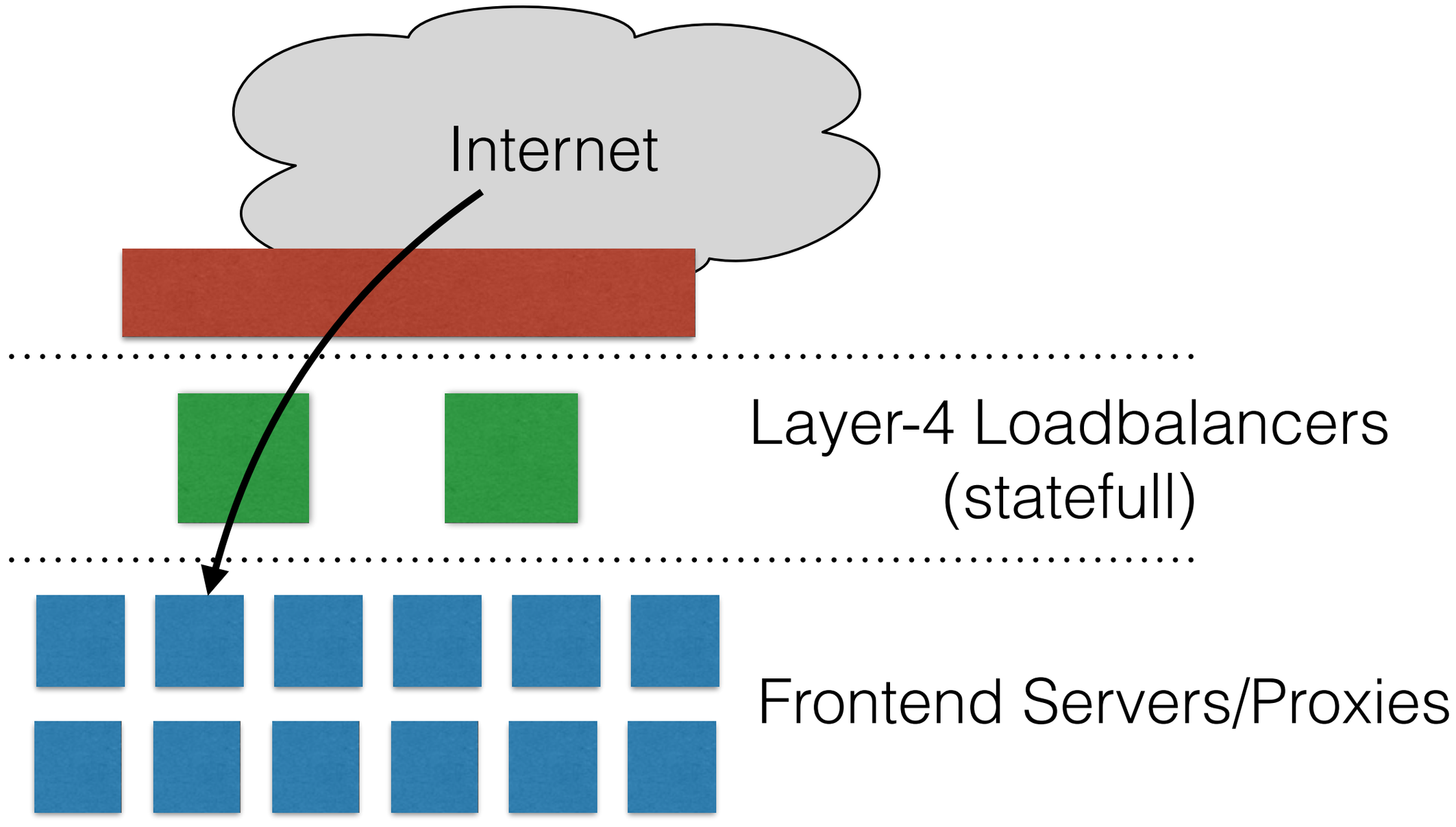
Adding/removing servers changes ECMP's hashing

Classical loadbalancer infrastructure



Statefull loadbalancer allows addition/removal of servers

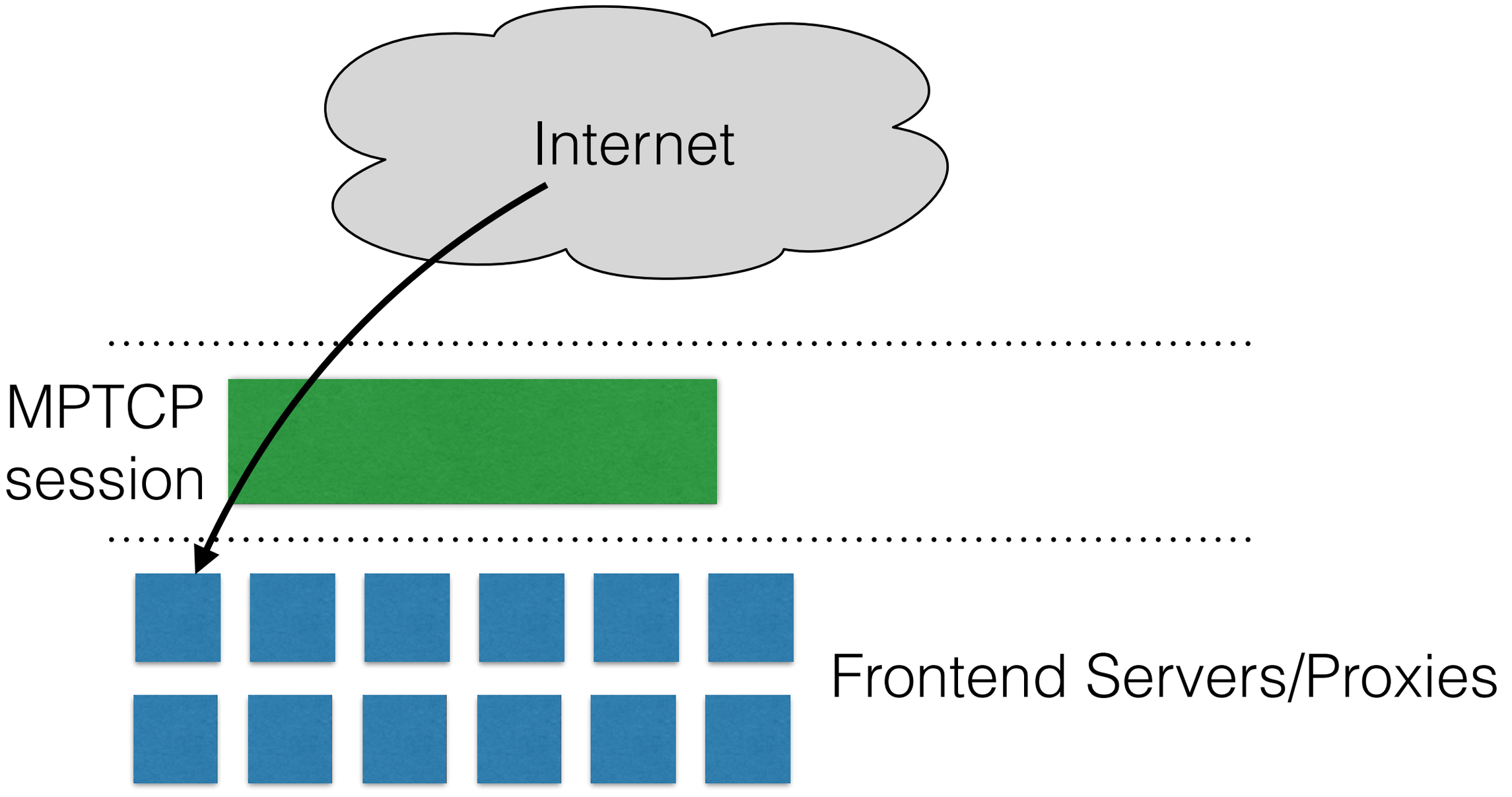
Classical loadbalancer infrastructure



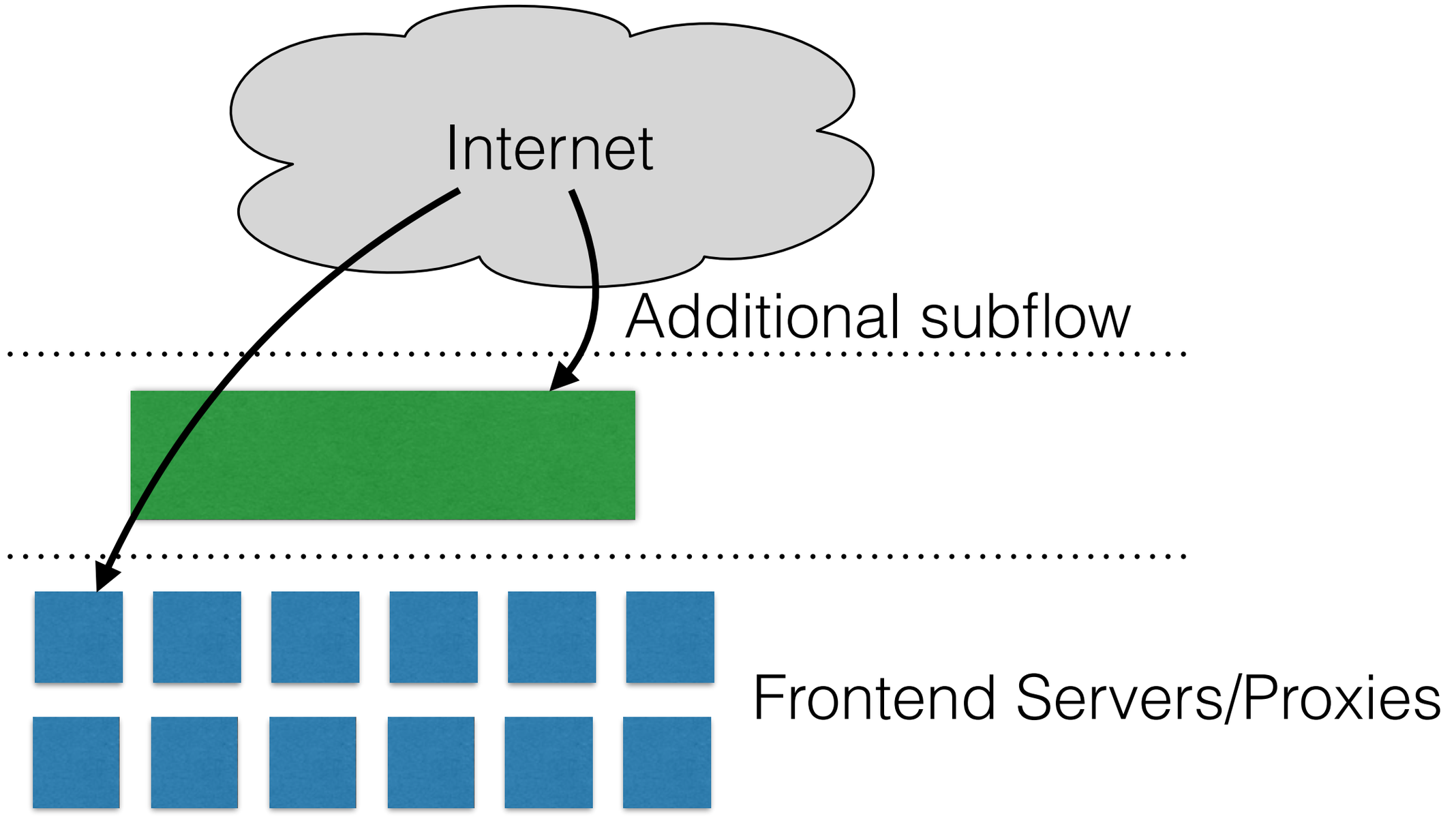
Scaling out to multiple loadbalancers

Problems with MPTCP

Single loadbalancer



Single loadbalancer



State-tracking & Token-collision

- ✓ Layer-4 loadbalancer needs to track <token-to-server> mapping

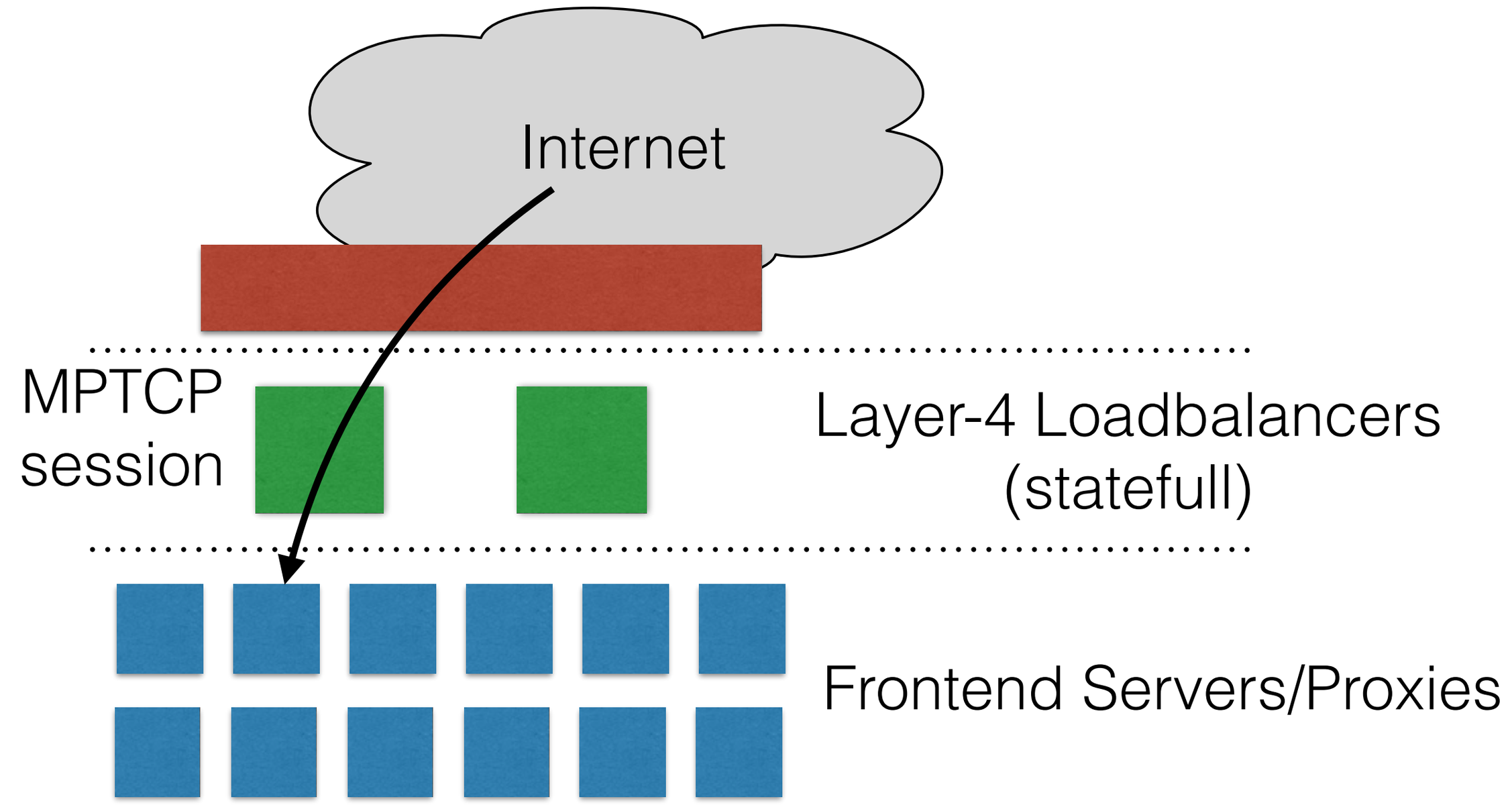
State-tracking & Token-collision

✓ Layer-4 loadbalancer needs to track <token-to-server> mapping

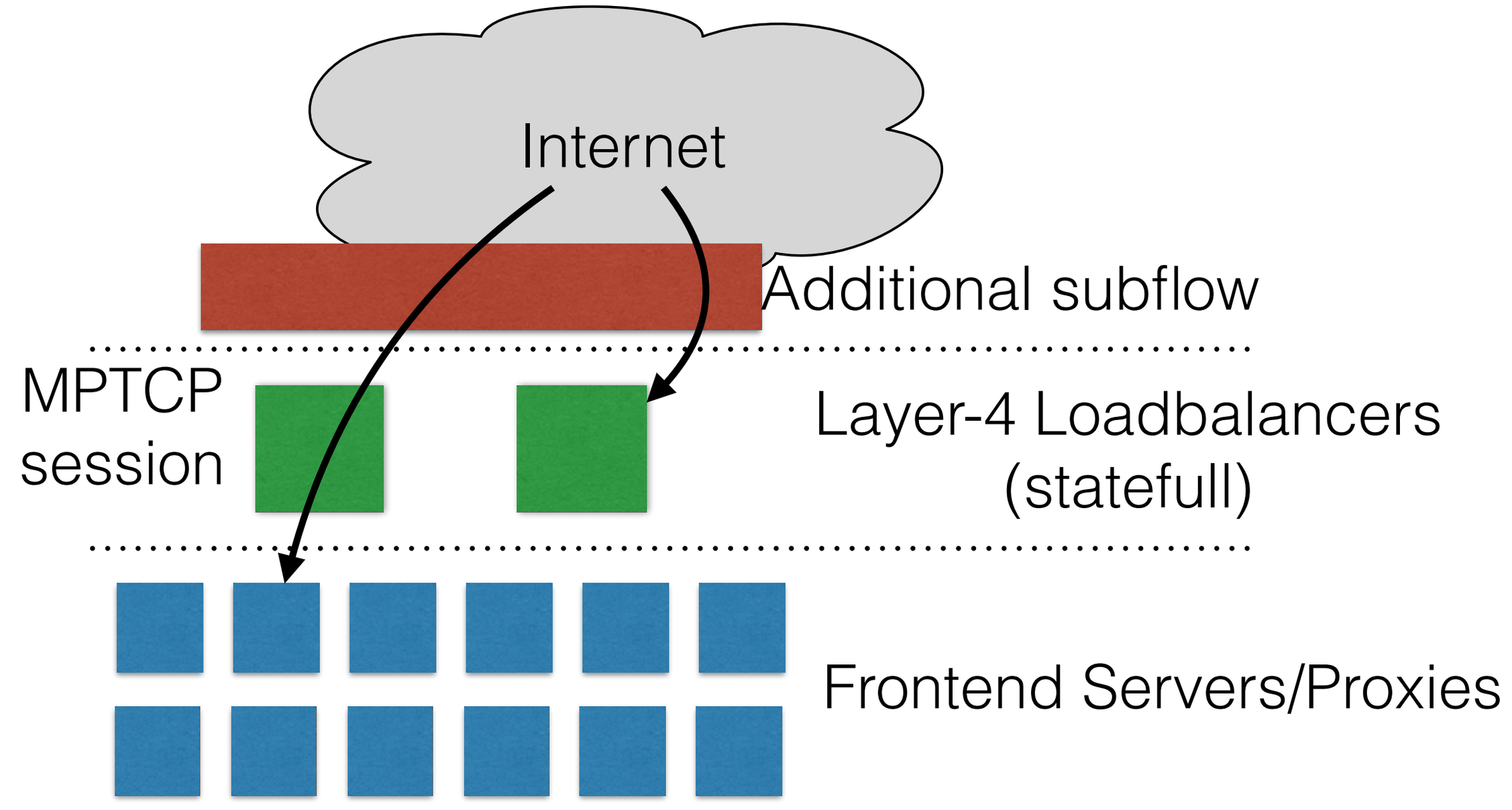
X Token collision on different servers are possible

- Loadbalancer cannot know which is the intended server (multiple MPTCP-sessions with same token)
- Servers would need to sync state (not feasible)

Multiple loadbalancers



Multiple loadbalancers



MPTCP-state synchronization

- Subflows belonging to the same MPTCP-session may reach different loadbalancers

  - MPTCP-state would need to be synchronized across loadbalancers (not feasible)

Solution-space

Underlying issue

- Issue is that tokens cannot carry any meaning (hash of the key)
    - Servers cannot guarantee uniqueness of token
    - Loadbalancer does not know where to forward a flow to

➡ Make token-generation locally meaningful

Shared secret token-generation

- Loadbalancers and servers share local secret Y
- Each server owns a range of integers (unique)
- token = encrypt(X, Y)
  with X an element of the server's range
- Loadbalancer does decrypt(token, Y) = X
  X indicates the server to forward this flow to

How to signal the token?

- Implicitly through different token-generation algorithm

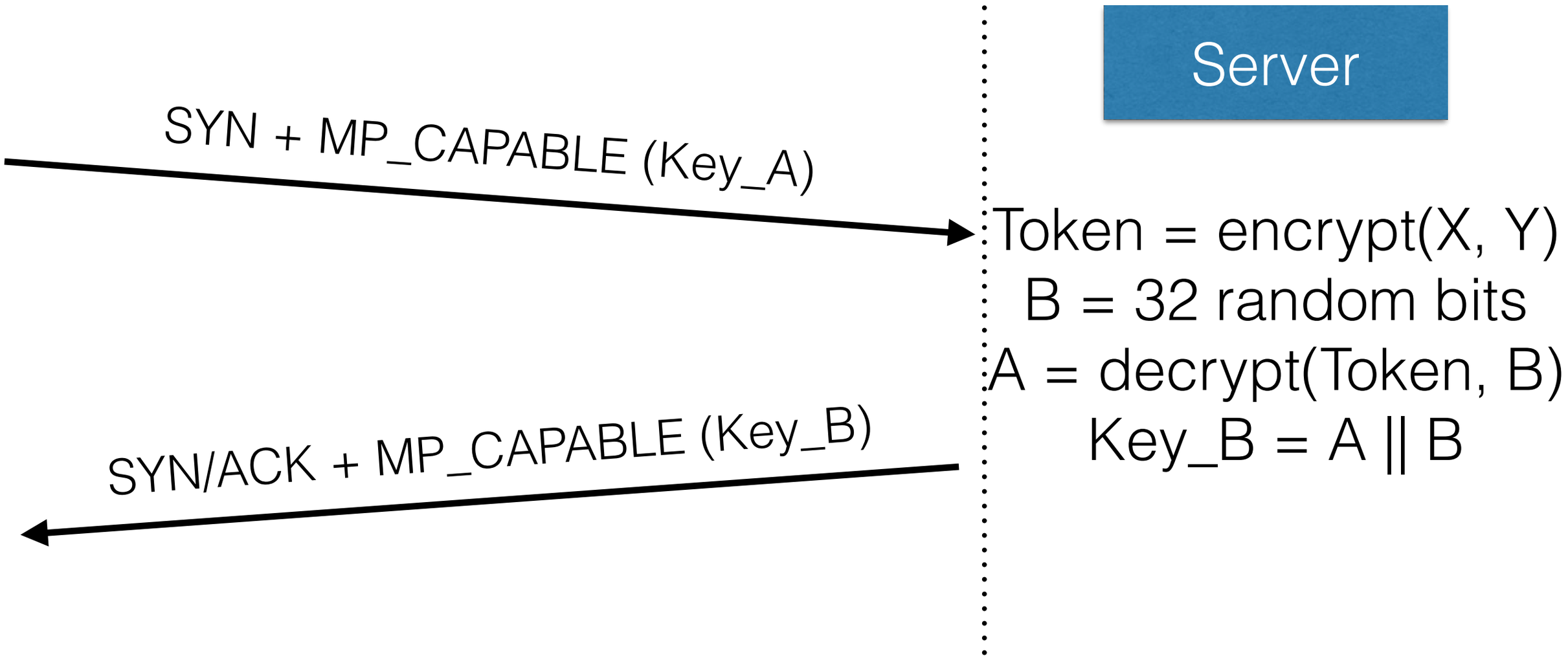
- Explicitly inside the MP_CAPABLE

Implicitly
through different token-generation

- MPTCP-key is 64-bit: 

- Token = block_cipher (A, B)

Implicitly
through different token-generation



Implicitly
through different token-generation

Pros & Cons:

✓ No wire-change required

X Token linked to the key

- Reduces entropy of the key by 32 bits

Explicitly
inside the MP_CAPABLE

SYN + MP_CAPABLE (token)

SYN/ACK + MP_CAPABLE (Key_B, token)

ACK + MP_CAPABLE (Key_A, Key_B)

(assuming deployment of draft-paasch-mptcp-syncookies)

Explicitly
inside the MP_CAPABLE

Pros & Cons:

✓ Token non-related to the key (entropy not reduced)

x TCP-option space issue (token consumes 4 more bytes)

- Might reduce Key_B to 32 bits and increase Key_A to 96 bits

x Server needs to act state fully if it wants to initiate connections to the client

Conclusion

- Token should be locally meaningful
  - ✓ Guarantees uniqueness
  - ✓ Enables distributed layer-4 loadbalancers
  - ✓ Enables large-scale deployment
- Signaling is still an open question