ICMP: The Problem (1)

```
nft add rule ip t c icmp sequence 3 counter
nft add rule ip t c icmp mtu 4 counter
nft add rule ip t c icmp gateway 5 counter
[...]
icmp sequence 3 counter packets 0 bytes 0
icmp sequence 4 counter packets 0 bytes 0
icmp id 0 icmp sequence 5 counter packets 0 bytes 0
```

ICMP: The Problem (2)

- ICMP protocol: Payload differs depending on type
- Field check is done via payload expression
- Return path doesn't know what user specified

ICMP: Questions

- Should `icmp id` be valid for (and match on) other types than echo-request and echo-reply?
- Implicitly create `icmp type` match or not?

ICMP: Ideas

1. Protocol CTX for ICMP type
   ⇒ `icmp id` is valid for multiple types
2. Make use of userdata to store meta info
3. Introduce print CTX to lookup earlier `icmp type` match from (same issue as with (1))

Echo: Usage in Scripts

```
# handle=$(nft --echo add rule ip t c counter)
# nft delete $handle
```

Echo: Current Implementation

```
# nft --echo add table ip t
table ip t
# nft --echo add chain ip t c
chain ip t c
# nft --echo add set ip t portset \
        '{ type inet_service; }'
set ip t portset
# nft --echo add element ip t portset '{ 80, 443 }'
element ip t portset { http }
element ip t portset { https }
# nft --echo add rule ip t c \
        tcp dport @portset drop
rule ip t c handle 2
```

Echo: Bugs

- Sets with interval flag

```
# nft --echo -nn add element ip t \
        portrange '{1-10}'
element ip t portrange { 0 }
element ip t portrange { 1 }
element ip t portrange { 11 }
# nft --echo -nn add element ip t \
        portrange '{20-30}'
element ip t portrange { 20 }
element ip t portrange { 31 }
```

- Robot argument: What if something recreates an item with same name?