DNS spoofing/redirecting with BIND's response policy zones

There was a talk on BIND's RPZ (response policy zones) feature at SwiNOG meeting last friday.

Fun story behind: I happened to implement the RPZ technology at my employer just days before, though with another target in mind.
RPZ is described as a way to implement DNS firewalls. Well, this can be used for the good, and the bad.

Here's another good use to this technology.

The idea behind implementing RPZ was to gently force customers into change the recursive name server IP addresses.
Historic infrastructure sometimes demand to get rid of some well-known services, and this in turn sometimes needs pressure if all means of contacting the customers have failed before ;-)

So the approach described here is to override any DNS replies and replac them with the destination IP address of a landing page.
In order to also reach several portals, a whitelist is supplied as well.

First add two dummy zones. One is acting as a whitelist zone, so you can define domain exclusions.
Make sure to not allow queries to the zone, you don't want anyone bugging around with your whitelist zone.

```
zone "dnswhitelist.anytld" IN {
        type master;
        file "master/dnswhitelist.anytld";
        allow-query { none; };
};
```

Should you run multiple name servers, you can use regular zone transfer mechanics. You may want to add an additional allow-transfer clause to the master statement above in order to restrict zone transfers.
Then setup the zone as slave zone accordingly, and point them to the master, as you would do with any authoritative zone.

Now for the contents of the whitelist zone.
This looks pretty much like any regurlar zone, you add a SOA and register a dummy name server (although the latter is not strictly needed).
To match on any (sub-)domains, just add them and point the entries to themselves.
Make sure that the left-hand RR doesn't have the final dot, while the right-hand data has.

```
;$ORIGIN dnswhitelist.anytld.
$TTL 86400

@               SOA localhost.dnswhitelist. hostmaster.dnssandbox. ( 2016103106 12h 15m 3w 2h)
                NS localhost
localhost       A       127.0.0.1


acme.net    CNAME   acme.net.
*.acme.net  CNAME   *.acme.net.
```

Now for the blacklist zone.
This looks fairly similar, and the same comments as above apply to a master-slave setup.

```
zone "dnssandbox.anytld" IN {
        type master;
        file "master/dnssandbox.anytld";
```

```
        allow-query { none; };
    };
```

As for the zone file contents, just register a wildcard RR pointing to the IP(v6) address of the landing page server.
You may also use a CNAME as redirection target. Mind however that the CNAME must be on the whitelist then.

```
;$ORIGIN dnssandbox.anytld.
$TTL 30

@           SOA localhost.dnssandbox. hostmaster.dnssandbox. ( 2016103107 12h 15m 3w 2h)
            NS localhost
localhost     A     127.0.0.1


*     A     192.0.2.1
```

To enable this in BIND, some final config lines are needed.

```
    response-policy {
        zone "dnswhitelist.anytld" policy passthru;
        zone "dnssandbox.anytld" policy given;
    };
```

The whitelist is registered using the **policy passthru** keyword, which causes all matching left-hand resource records to be passed as is, without rewriting.Additionally this also means that the target's destination (the right-hand data part in the zone file) is totally ignored. You could use the **policy given** keyword of course, however then you would need to take care that the redirection target is set properly on the right-hand side.
If you want an easy-to-use whitelist, whithout bothering for syntax, just use the **passthru** directive.

The second zone, the actual sandbox, is registered using the **policy given** keyword. This means that the policy is handed in from the zone file from the right-hand data value.
So by registering the redirection target to an IP address as shown above, all requests will be set to that value as a response.

You can provide further policies inside the zone by stating the proper syntax in the data section, see also in BIND administrators manual.

One more thing: Can this configuration also be applied selectively?
Short and simple: Yes, using views and ACLs.

So assuming that the same instance must provide different view scopes, and thus behave differently to clients depending on their source, here's how to do it.
This example applies the response policy to all clients coming from 192.0.2.128/25 network, whereas everyone else could be served regularly.

```
acl "access_clients" {
 192.0.2.128/25;
};

view "dnssandbox" {
    match-clients { access_clients; };
    response-policy {
```

```
        zone "dnswhitelist.anytld" policy passthru;
        zone "dnssandbox.anytld" policy given;
    };

    zone "dnswhitelist.anytld" IN {
        type master;
        file "master/dnswhitelist.anytld";
        allow-query { none; };
    };

    zone "dnssandbox.anytld" IN {
        type master;
        file "master/dnssandbox.anytld";
        allow-query { none; };
    };
};

view "global" {
    match-clients { any; };

    [[ other zone statements and/or other rpz policies go here ]]
};
```