

### Brief History

- I wanted some feature in qmail-smtpd
- First version released
   in 2001 in production since before then
- v0.10 with the modular features a year later
- Used for all inbound mail at perl.org, apache.org, cpan.org and many others (corporations small and large)
- MIT licensed! (makes all other licenses look cumbersome and restrictive!)



# Major Feature: Plug-ability



- Minimal feature set in the "core"
- New Perl programmers make their first plugin in an afternoon
- Experienced ones in minutes!

#### My first plugin! (Be friendly to "human users")

```
sub hook_quit {
my $qp = shift->qp;
# if she talks EHLO she is probably too
# sophisticated to enjoy the fun, so skip it.
return (DECLINED) if ($qp->connection->hello || '') eq "ehlo";
my $fortune = '/usr/games/fortune';
return DECLINED unless -e $fortune;
my @fortune = `$fortune -s`;
$qp->respond(221, $qp->config('me')
                   " closing connection.", @fortune);
return DONE;
```

## Check basic headers

```
sub hook data post {
my ($self, $transaction) = @
 return (DENY, "You have to send some data first")
   if $transaction->data size == 0;
 return (DENY, "Mail with no From header not accepted here")
   unless $transaction->header->get('From');
my $date = $transaction->header->get('Date');
 return (DENY, "Mail with no Date header not accepted here")
  unless $date;
# check that the date header is "within a few days of now"
 return (DECLINED);
```

#### Enforce "server says hello first" protoco

(and reject spam and viruses) use IO::Select; sub register {  $my (\$self, \$qp) = @_;$ \$self->register hook('connect', 'connect handler'); sub connect handler { my (\$self, \$transaction) = @ ; my \$in = new IO::Select; \$in->add(\\*STDIN) || return DECLINED; if  $(\sin-> can read(1))$  { \$self->log(LOGDEBUG, "remote host talked before we said hello"); return (DENYSOFT, "Don't be rude and talk before I say hello!"); \$self->log(LOGINFO, "remote host said nothing spontaneous, proceeding"); return DECLINED;

## Save incoming mail to DB

```
sub hook queue {
my ($self, $transaction) = @ ;
my $dbh = DBI->connect cached(...);
 $dbh->do(q[insert into mails
            (msg id,header,body)
             values (?,?,?)], undef,
    $transaction->header->get('Message-Id'),
    $transaction->header->as string,
    $transaction->body as string
return (OK, "Queued! $msg_id");
```

## More Plugins!

greylisting

spamassassin

clamd

dnsbl

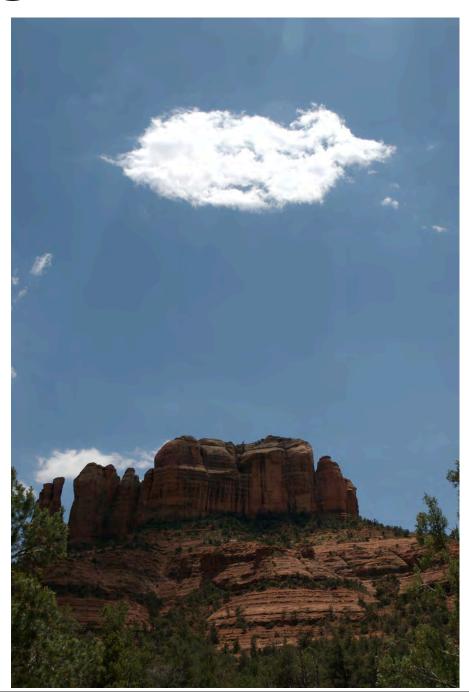
authentication

tls (transport level security = ssl)

check OS of connecting client

domainkeys

and many more





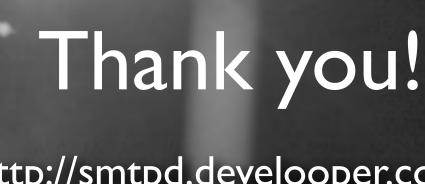


## Transports

 Start a process per connection (CGI model, the original 'qpsmtpd' program)



- Start the daemon and fork on demand ('qpsmtpd-forkserver')
- Start the daemon and prefork (like Apache!) ('qpsmtpd-prefork')
- Apache::Qpsmtpd (run under mod\_perl!)
- Event based, handle 1000s of concurrent connections. Hello spammers, we can cope! ('qpsmtpd-async')



http://smtpd.develooper.com/ask@develooper.com

www.yellowbot.com