ApacheCon Europe

Sinsheim, Germany 5th-8th November 2012

What's New in Apache HTTP Server 2.4

Rainer Jung
kippdata informationstechnologie GmbH

Agenda

- Warm Up
- Version Overview
- MPMs
- Core: Logging, Expression Parser, Define
- Modules
- Community
- Discussion!

Agenda

- Warm Up
- Version Overview
- MPMs
- Core: Logging, Expression Parser, Define
- Modules
- Community
- Discussion!

Warm Up – Who am I?

- Who am I?
  - Main Experience: Systems Integration
    - Adding production qualities (performance, availability) to applications which are usually thought to be finished
    - Troubleshooting (through all layers of the software stack)

and on top of this a typical Open Source career ...

Warm Up – Who am I?

- Open Source Reader
  - Problem analysis
- Open Source Contributor
  - Problem Fixing
- Committer and PMC Member
  (Project Management Committee)
  - Apache HTTP Server
  - Apache Portable Runtime (APR)
  - Apache Tomcat and mod_jk
- Apache Software Foundation Member

Warm Up – Who are you?

- Who are you?
  - Who belongs to
    - Development?
    - Operations?
  - Who uses
    - Apache 2.4?
    - Apache 2.2?
    - Apache 2.0?
    - Apache 1.3?
  - Who uses other web servers than Apache in production (Nginx, Lighttpd, IIS)

Warm Up – Who are you?

- Who are you?
  - Who runs a top load of more than 1000 requests per second?
  - Who did already contribute to an Open Source project?
    - Patch, Documentation, Problem analysis
  - Who is a member of an Open Source project?

Agenda

- Warm Up
- Version Overview
- MPMs
- Core: Logging, Expression Parser, Define
- Modules
- Community
- Discussion!

Version Overview of the Apache Web Server

- Apache 2.0: 2000, current: 2.0.64 (October 2010!)
    - stable since 2.0.35, April 2002
- Apache 2.1: 2003
    - old development branch
- Apache 2.2: December 2005, current: 2.2.23 (13.09.2012)
- Apache 2.3: 2008
    - old development branch, 16 „releases“, of which were 5 betas
- Apache 2.4: branched in November 2011
    - 2.4.0 not released, 2.4.1 first GA Release February 2012
    - current 2.4.3 (21.08.2012), probably 2.4.4 before end of 2012

Version Overview of the Apache Web Server

- User Adoption
  - Very slow for 2.0
  - Much better for 2.2
  - Looks promising for 2.4
    - to early for serious estimates about adoption
    - a lot of feedback on the users discussion list

Most Important New Features in Apache 2.4

- The most important new features (overview)
  - Better scalability by using asynchronous processing (Event MPM)
  - Improved logging (Access Log, Log Levels, Debugging)
  - Consistent and improved use of expressions in configuration (Expression Parser, If Directive, Define)
  - Reworked AAA configuration
  - Reduced memory consumption
  - Improvements in mod_ssl, mod_proxy, …
  - About 40 new modules in total

Agenda

- Warm Up
- Version Overview
- MPMs
- Core: Logging, Expression Parser, Define
- Modules
- Community
- Discussion!

Multi Processing Modules – Overview

- Multi Processing Module (MPMs)
  - Handle the mapping of TCP connections and HTTP requests to processes and threads
- Implemented as modules because of
  - platform dependence
  - possibility to adopt to special use cases
  - more choices using 3rd party MPMs
- Interaction with parent process (start, stop, restart)
- Query interface (retrieve MPM config and status)

Multi Processing Modules – Prefork MPM

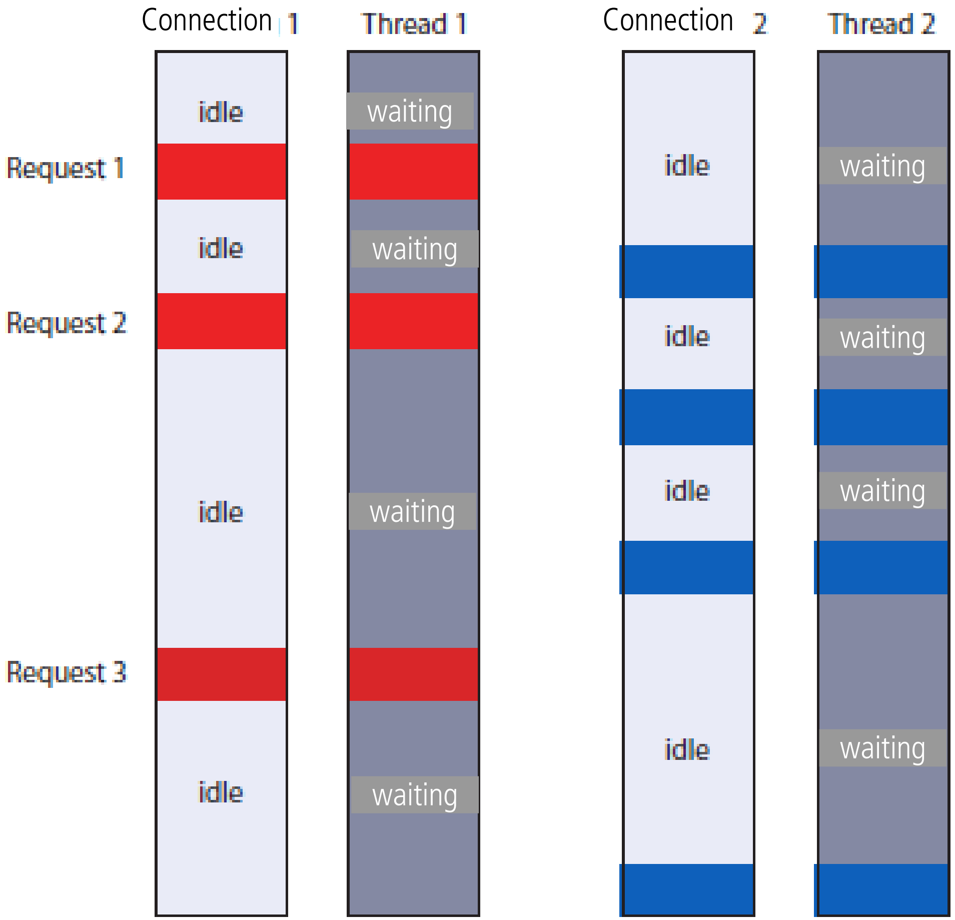
- Prefork (Unix/Linux) – Exists since 1.3
  - simple design
    - single-threaded, one process per TCP connection
    - scales by adding processes
    - limited scalability for huge numbers of connections
      - memory as a bottleneck
      - Example: download server
  - fast (short code path, no locking)
  - stable (nice problem separation, e.g. in case a module crashes)
  - good choice for modules which are not thread-safe

Multi Processing Modules – Worker MPM

- Worker (Unix/Linux) – Alternative since 2.0
  - more complex design
    - multi-threaded
    - one thread per TCP connection
    - also scales by adding processes
      - each process contains the same fixed number of threads
    - scales better for huge numbers of connections
  - still fast, some locking overhead
  - reduced problem separation, a crash stops several requests
  - only for modules which are thread-safe

Multi Processing Modules – Worker MPM

▪ Worker Visualized



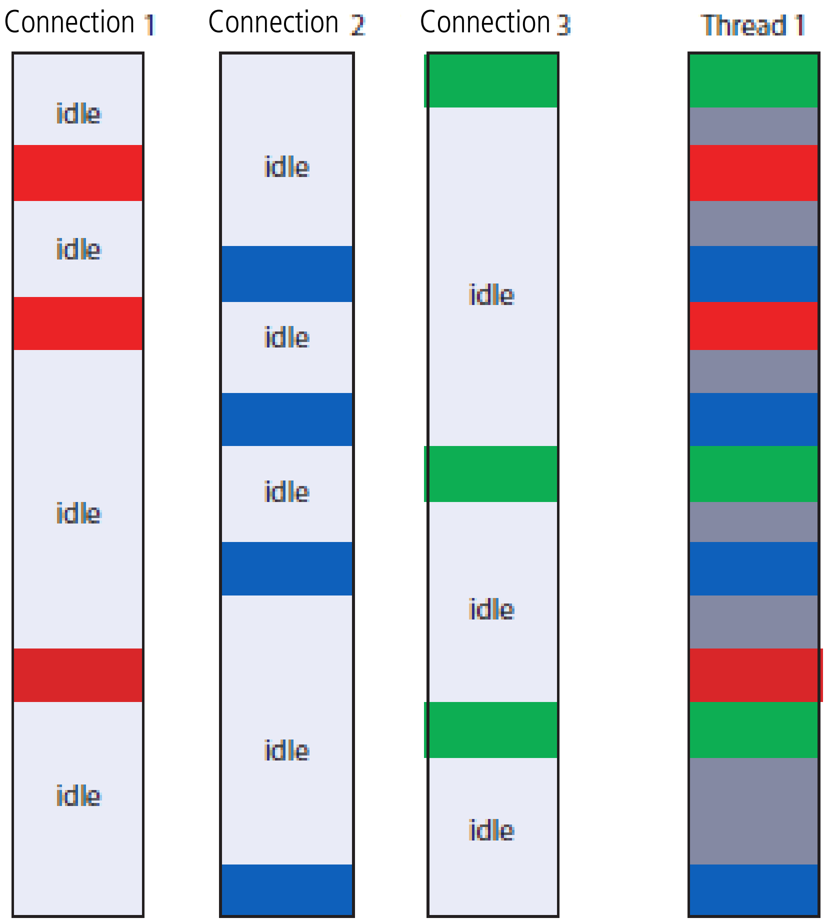
Graphics: Oliver Diedrich, Heise Verlag

Multi Processing Modules – Event MPM

- Event (Unix/Linux) – Alternative since 2.2
    - complex design
        - multi-threaded
        - Event based architecture, more details following soon
        - supports much more TCP connections with the same thread count
        - also scales by adding processes
            - each process contains the same fixed number of threads
        - scales much better for huge numbers of connections
    - more context switches
    - reduced problem separation
    - only for modules which are thread-safe

Multi Processing Modules – Event MPM

- Event Visualized



Graphics: Oliver Diedrich, Heise Verlag

Multi Processing Modules – Other MPMs

- Other platforms
  - winnt (Windows)
    - multi-threaded
    - one thread per TCP connection
    - only one worker process, using a fixed thread count
  - mpmt_os2 (OS/2)
  - netware (Netware)
- 3rd-party MPMs
  - Example: mpm-itk

Multi Processing Modules – New and removed MPMs

- In 2.2
    - Removed the experimental MPMs „leader“, „perchild“, „threadpool“
    - „Event“ as a new „experimental“ MPM (Unix/Linux)
- In 2.4
    - Removed MPMs for BeOS, TPF, A/UX, Next, Tandem
    - MPM „Event“ no longer experimental, instead Default!
- In trunk
    - New experimental MPM „simple“

Multi Processing Modules – Event APIs

- Event based architecture for the Event MPM
  - Events on connections are monitored by a listener thread and dispatched to worker threads
  - Epoll (Linux), Kqueue (BSD), Event Ports (Solaris)
    - More efficient than select and poll, especially if there are
      - huge poll sets (many connections)
      - high rates of changes for the poll sets (many events)
    - In 2.2 „experimental“, so only few users, source code only slowly getting mature
      - Solaris 10 GA: Kernel crashes during Apache stress testing on Niagara
      - exact API contracts weren't clear at the beginning

Multi Processing Modules – HTTP Keep Alive

- HTTP Keep Alive
  - Keep Alive means reusing a connection for multiple consecutive HTTP requests
    - RFC 2616 suggest to not use more than 2 connections to the same server from a single client (well behaved)
    - Keep Alive helps to lower latency, pages load more smoothly
    - Because of AJAX the 2 connections rule is frequently ignored
  - New requirement: web servers need to be able to handle lots of mostly idle connections efficient
    - Rule of thumb: Keep Alive increases connection count by factor 5x – 10x depending on the KeepAliveTimeout

Multi Processing Modules – Event and Keep Alive

- Event and HTTP Keep Alive
  - Event releases the worker thread, as soon as the HTTP response was sent
  - during Keep Alive (waiting for the next request) no worker thread will be used or blocked
    - fewer threads needed, scales much better
    - needs more context switches
  - When using SSL it still uses one thread per connection
  - The Event MPM already exists for 2.2
    - But was improved and stabilized a lot for 2.4

Multi Processing Modules – Event in 2.4

- New in Event for 2.4
  - Event now is the default MPM on Unix/Linux
  - Async Write Completion
    - Threads are released, as soon as the response content is available but the network blocks when trying to write to the client (slow connections)
    - Example: download farm
  - Better sizing behavior and monitoring than in 2.2
  - Many bugs fixed

Multi Processing Modules – About Protocols

- The need of higher connection scalability
  - Problem to solve: server should send messages to the clients (instead of request-response)
    - But: Clients do not accept connections from the server
    - Idea: tunneling through client to server HTTP
      - Client opens an HTTP connection
      - Server and Client agree to keep the connection open instead of sending a reponse
      - Now they proceed with a proprietary protocol on this connection
    - Result: lots of concurrently open connections, even if idle for a long time
    - Buzzwords: Hanging HTTP, Long poll, Comet

Multi Processing Modules – About Protocols

- Example: all customers with mobile phones should continuously keep a connection to the server open
  - Problem of scaling over an enormous number of open but mostly idle connections
  - Problem of sending messages in a fair way (concurrent)
- Java Servlet 3 Spec contains a few early parts of standardization (async API in a servlet container)
- Event architecture in gateways / reverse proxies gets essential in the near future

Multi Processing Modules – About Protocols

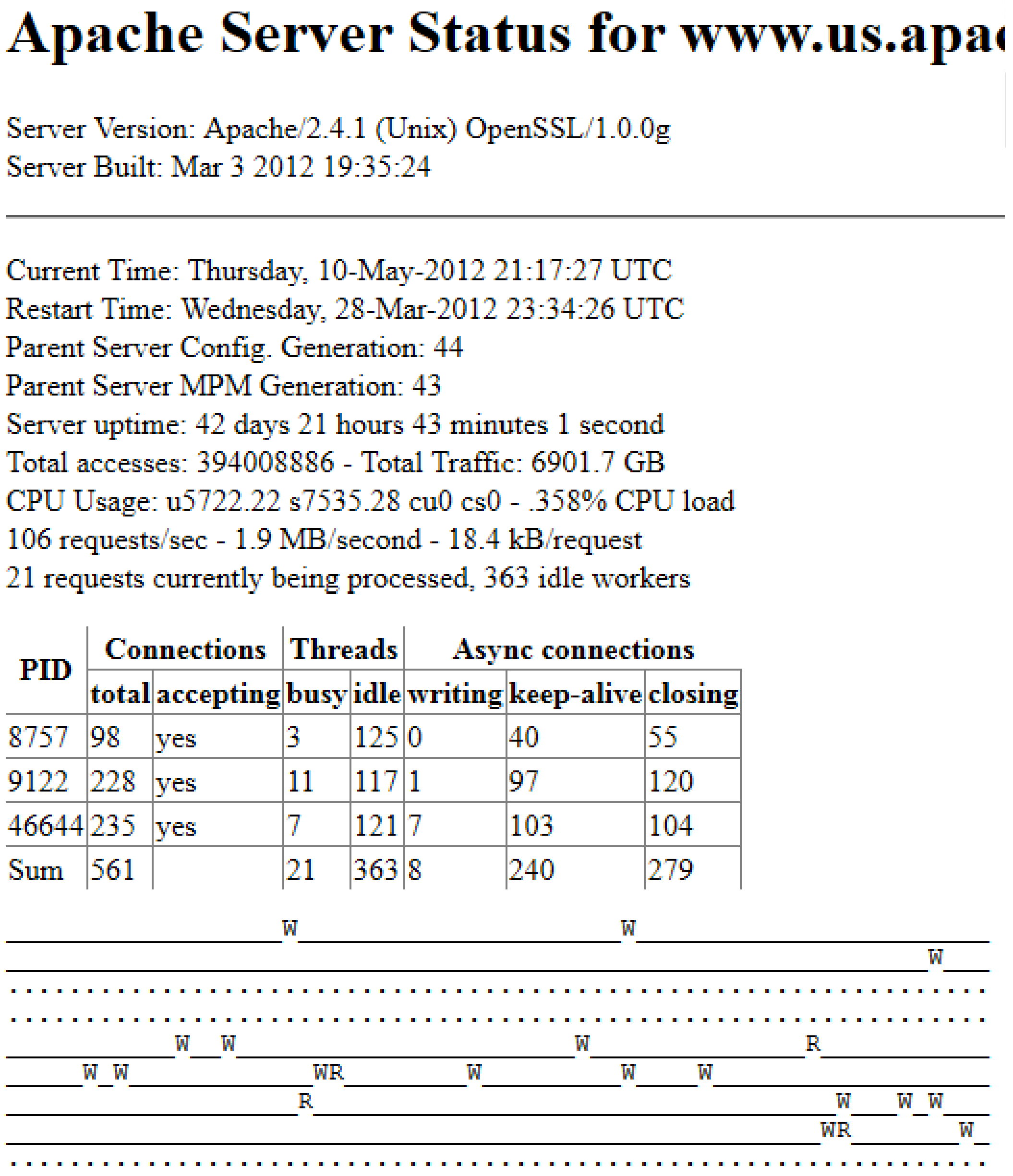
- Protocol standardizations are on the way
  - Strong support from browser vendors
    - HTML5 WebSockets at W3C (http://dev.w3.org/html5/websockets/)
  - Much interest by game developers
  - IETF: BiDirectional or Server-Initiated HTTP (hybi)
    - RFC 6455 Proposed Standard: http://datatracker.ietf.org/wg/hybi/
  - Google SPDY: http://www.chromium.org/spdy
  - HTTPbis: Agenda to clarify HTTP/1.1, work on HTTP/2.0
    - http://datatracker.ietf.org/wg/httpbis/

Multi Processing Modules – Aspects of Event

- Interesting aspects of Event
  - How does one achieve a good MPM sizing?
    - We can configure the maximum of the total thread count
      - It used to be called „MaxClients“, now the better name is „MaxRequestWorkers“
      - There is no direct limit for the number of connections per process
    - How many threads can handle how many connections?
      - Default: each Process only accepts new connections, if: #idleConnections < 3 * #idleThreads
      - The factor „3“ is configurable

Multi Processing Modules – Aspects of Event

- Interesting aspects of Event
  - Scoreboard enhanced for better Monitoring/Sizing
    - New counters for asynchronous connections
    - Status of the „Connection Throttling“ („accepting“)



Apache Server Status for www.us.apa

Server Version: Apache/2.4.1 (Unix) OpenSSL/1.0.0g
Server Built: Mar 3 2012 19:35:24

Current Time: Thursday, 10-May-2012 21:17:27 UTC
Restart Time: Wednesday, 28-Mar-2012 23:34:26 UTC
Parent Server Config. Generation: 44
Parent Server MPM Generation: 43
Server uptime: 42 days 21 hours 43 minutes 1 second
Total accesses: 394008886 - Total Traffic: 6901.7 GB
CPU Usage: u5722.22 s7535.28 cu0 cs0 - .358% CPU load
106 requests/sec - 1.9 MB/second - 18.4 kB/request
21 requests currently being processed, 363 idle workers

PID	Connections		Threads		Async connections		
	total	accepting	busy	idle	writing	keep-alive	closing
8757	98	yes	3	125	0	40	55
9122	228	yes	11	117	1	97	120
46644	235	yes	7	121	7	103	104
Sum	561		21	363	8	240	279

```
__________________W_____________________W_______________________
____________________________________________________________W___
................................................................
................................................................
___________W__W______________________W______________R___________
_____W_W____________WR________W_________W____W__________________
___________________R__________________________________W___W_W___
_____________________________________________________WR_______W_
................................................................
```

Multi Processing Modules – Aspects of Event

- More interesting aspects of Event
  - No asynchronous handling of requests and connections when using SSL
  - The same number of threads can handle more connections, but still each connection needs a file descriptor. So we have fewer processes but each process will need more file descriptors.

Multi Processing Modules – MPMs as Dynamic Modules

- New in 2.4: MPMs as dynamically loadable modules
  - MPMs now can be build as dynamically loadable modules
    - Example: „configure –enable-mpms-shared=all“
  - It eases distributing Apache with runtime exchangeable MPM
  - Load the MPM using LoadModule
    - LoadModule mpm_event_module modules/mod_mpm_event.so

Agenda

- Warm Up
- Version Overview
- MPMs
- Core: Logging, Expression Parser, Define
- Modules
- Community
- Discussion!

Core: Logging – ErrorLog

- Error Logging
  - New Default Format:
    - sub second timestamp precision (microseconds)
    - module and log level
    - process ID and thread ID (which log lines belong to the same request)
    - Each message in the ErrorLog contains a unique message token
  - the log format for the ErrorLog is now configurable!
    - Client-IP, Request-URL, User-Agent, Referer, …

Core: Logging – ErrorLog

- Examples from the Error-Log

  -
    ```
    [Fri May 11 01:03:15.039653 2012] [mpm_event:notice]
    [pid 5053:tid 1] AH00489:
    Apache/2.4.2 (Unix) configured -- resuming normal operations
    ```

  -
    ```
    [Fri May 11 01:08:02.035185 2012] [core:notice]
    [pid 5053:tid 1] AH00094:
    Command line: '/path/to/apache/bin/httpd'
    ```

  -
    ```
    [Fri May 11 01:08:03.715738 2012] [core:info]
    [pid 5069:tid 16] [client 127.0.0.1:63608] AH00128:
    File does not exist: /path/to/apache/htdocs/secret
    ```

Core: Logging – ErrorLog

- Error Log Levels
  - New levels TRACE1 to TRACE8
  - LogLevel configurable per module
    - Example: „LogLevel warn authz_core:debug core:trace4“
    - Separate log files removed (RewriteLog/RewriteLogLevel, ...)
  - LogLevel configurable in Directory/Location and in „If“
    - detail logging for special URLs
  - Log implementation checks whether log level is active
    - no message formatting if log level is not active
  - Lots of new Trace log messages in the core!

Core: Logging – ErrorLog

- More about Error Logging
  - Correlation ID for connections and requests, allow to correlate easily between ErrorLog and AccesLog
    - Add the ID using „%L“
  - New log line printed at start contains the start command line
  - Add individual log messages using mod_log_debug
    - Ex.: LogMessage "IPv6 timeout from %{REMOTE_ADDR}" "expr=-T %{IPV6} && %{REQUEST_STATUS} = 408"
    - Can be also used in Directory/Location

Core: Logging – AccessLog

- Timestamps in the AccessLog
  - Adding subsecond timestamps (milliseconds or microseconds)
  - Logging Unix milli- or -microseconds
  - Choosing between request start and reponse end
    - everything can be mixed and will be consistent

… who currently uses %D in the LogFormat?

Core: Expression Parser

- Expressions in the configuration
  - Unifying the possible expressions in
    - RewriteCond
    - SetEnvIfExpr (new)
    - <If *expression*> (new)
    - Require (new)
  - well documented
  - <If *expression*>: new container directive
    - analogous to Location, Directory, Files

Core: Expression Parser

- Expressions in the configuration – More
  - Much more powerful, everything works everywhere
    - *word* in *wordlist*
    - Regular expression */regexp/* oder */regexp/i*
    - String matches *-ipmatch*, *-strmatch*, *-strcmatch*, *-fnmatch*
    - Functions
      - retrieve headers using *req* (Request), *resp* (Response)
      - retrieve table entries using *reqenv*, *osenv*, *notes*
      - convert string case using *tolower*, *toupper*
      - encode strings with percent encoding using *escape*, *unescape*
    - and much more!

Core: Expression Parser

- Expression Examples
  - Goal: Detect IP in network 195.226.29.0/25
    - Before 2.4 only possible in Allow/Deny
    - Everywhere else: „^195\.226\.29\.([0-9][0-9]?$|1([01]|2[0-7]))“
  - New: where ever the expression parser is allowed:
    - -R '195.226.29.0/25'
  - Example:

```
<Location /internal>
  <If "-R '195.226.29.0/25'">
    Options +Indexes
  </If>
  <Else>
    Require all denied
  </Else>
</Location>
```

Core: Define

- Integration of mod_define into the Apache core
  - Module for Apache 1.3 contained in mod_ssl distribution
- Directive „Define name [value]“
  - Defines Variable with name „name“ (like „-Dname“)
    - Usable in <IfDefine name>...</IfDefine>
  - By using „value“ the variable gets a value assigned
    - Can be referenced in arbitrary places in the configuration using the syntax ${name} !
  - Already worked in 2.0 using OS environment variables
    - But using „Define“ you can change the values gracefully

Agenda

- Warm Up
- Version Overview
- MPMs
- Core: Logging, Expression Parser, Define
- Modules
- Community
- Discussion!

Modules – API

- API Compatibility
  - Usually a rebuild of modules which worked with 2.2 should suffice
    - Notable exceptions: Scoreboard, AAA, IP Split
  - Small adjustments can easily improve the modules, e.g. replacing one macro call in order to use the new log configurability
  - Many modules are already 2.4 ready
    - mod_jk, mod_security, mod_php, mod_macro, ...

Modules – Build

- Building Modules
  - Modules are now build by default as dynamically loadable (DSO).
  - Optionally they can still be build statically linked into the httpd binary
  - Pre cooked module sets are „none“, „few“, „most“, „all“, „reallyall“
    - Default is „most“
  - Not all modules get enabled by default! (LoadModule partially commented)
    - For test purposes: configure flag „--enable-load-all-modules“

Modules – Scripting

- Scripting in HTTPD – dynamic configurations
  - Problems:
    - How to configure policies/rules
      - regular expressions and back references in mod_rewrite
    - How to build if-then-else constructs
      - complex list of RewriteRule
    - How to do configuration depending on request data (Client, IP, …)
      - environment variables, mod_rewrite

The results are often hard to understand and maintain.
The available configuration syntax is too limited.

Modules – mod_lua

- Old idea: mod_perl
  - But: Perl interpreter is to heavy when running under high load, does not scale well
- New idea: mod_lua
  - light weight
  - optional JIT compiler
  - written in ANSI-C, available for all platforms
  - excellent bindings from C to Lua and from Lua to C
  - Domain Specific Language can be build on top of Lua

Modules – mod_lua

- Some Lua background information
  - created 1993 in Brasil
  - stripped interpreter size only about 140 KB
  - Used as a plugin language e.g. in
    - Adobe Lightroom
    - World of Warcraft
    - Whireshark, Snort, VLC

Modules – mod_lua

- Lua interpreter lifecycle in Apache
  - LuaScope: once, request, conn, server
    - server: Pool of interpreter instances (reused)
- Script Caching
  - LuaCodeCache stat|forever|never
- Apache request data available from inside Lua
  - almost anything readable: uri, host, params, headers, notes, parse POST bodies, …
  - most things writable: uri, user, content_type, status, filename, headers (in/out), notes

Modules – mod_lua

- Lua scripts as a handler
  - handlers provide the response content
    - LuaMapHandler uri-pattern /path/to/script.lua [function-name]
    - Ex.: LuaMapHandler /(\w+)/(\w+) /scripts/$1.lua handle_$2
- Lua scripts as hooks
  - hooks influence the request processing
  - functionality of most modules is implemented in hooks
- Authorization Providers can be implemented in Lua
- Logging: r:info("This is an info log message") etc.

Modules – mod_lua

- Note: mod_lua is experimental
  - The API is not yet final, so it is possible that current configurations and scripts need to be adjusted for future 2.4 versions.

Modules – AAA

- AAA: Authentication, Authorization, Access Control
  - Improved separation of these aspects
  - Compatible with 2.2 way of configuring AAA by using the optional module mod_access_compat
  - Types of authentication (AuthType)
    - still basic and digest
  - Provider of data (AuthBasicProvider, AuthDigestProvider)
    - still anon, dbd, dbm, file, ldap
  - Authorization (Require)
    - still dbd, dbm, groupfile, host, user, owner, ldap

Modules – AAA Require

- AAA: Require
  - Require ip *address*, Require not ip *address*
  - Require host *domain_name*, Require local
  - Require all denied, Require all granted
  - Require env *let_me_in*, Require expr *expression*
  - Require valid-user
  - Require user *rjung*, Require group *admin*
  - Require file-owner, Require file-group
  - Require ldap-* (viele), Require ssl, Require method *METH*

Modules – AAA Require Container

- AAA: Require-Container
  - <RequireAll> <RequireAny> <RequireNone>
  - mod_authn_socache
    - Caching of authorization results of the providers dbd, dbm and file

Modules – mod_proxy

- mod_proxy in 2.2
  - mod_proxy_balancer and mod_proxy_ajp already exist since 2.2
    - Documentation partially hidden inside the mod_proxy page
  - Who already uses mod_proxy_balancer?
  - Who already uses mod_proxy_ajp?
  - Who uses mod_proxy?

Modules – mod_proxy

- mod_proxy in 2.4
  - ProxyErrorOverride: in case of error, use the error pages defined for Apache instead of the original back end response (backported to 2.2.23)
  - ProxyPass[Reverse]: now forbidden in <Directory> and <Files> (but still OK in <Location>)
  - Connections to the back end will be released as soon as the complete response was received
    - Even if sending the response to the client is not yet completed

Modules – mod_proxy

- mod_proxy in 2.4
  - New in mod_proxy_balancer
    - Balancer algorithm pluggable
      - Bundled: „byrequests“, „bytraffic“, „bybusyness“ and „heartbeat“. I will come back to „Heartbeat“ soon
      - Easy to write a proprietary balancer plugin
    - Enhanced Manager GUI
    - No longer uses scoreboard memory, instead uses mod_slotmen
      - As a result we can do the following:

        New members can be added to an existing load balancer dynamically via the GUI

Modules – mod_proxy

- mod_proxy in 2.4
  - mod_proxy_fcgi
    - Do not use, if the FastCGI process is single-threaded
    - Process manager not integrated, instead use bin/fcgistarter
    - Alternative: http://httpd.apache.org/mod_fcgid/
  - mod_proxy_ftp

Modules – mod_proxy

- Experimental stuff in mod_proxy in 2.4
  - „ping“ support via HTTP 100-Continue
    - Goal: Detect race conditions when using HTTP Keep Alive
  - mod_proxy_fdpass
    - Hand over the client socket via a Unix Domain Socket to some external process
      - ProxyPass /some/url fd:///path/to/my/unix/sock
      - the external process handling the unix domain socket directly sends the answer to the client without involving Apache again

Modules – Content Filters

- mod_substitute
  - Since 2.2: simple search and replace operations applied to the response content
- mod_sed
  - New in 2.4: full sed implementation
  - input- and output-Filter, so also allows request body rewriting
- mod_proxy_html
  - XHTML based transformations, e.g. link fixing

Modules – mod_proxy

- When a proxy sits in front of Apache: mod_remoteip
  - gets original client IP out of a request header
  - Uses resulting IP everywhere instead of the proxy IP
    - Authentication, Logging, ...
  - Header name and list of trustworthy proxies is configurable

Modules – mod_ssl

- mod_ssl in 2.4
  - Supports OCSP (Online Certificate Status Protocol) and OCSP Stapling
  - Directive „SSLFIPS On“activates FIPS mode in the SSL library
  - Session cache implemented on top of standard module mod_socache_*
    - Example: mod_socache_shmcb
    - Alternatives: memcache, distcache, dbm
    - socache_* module must be loaded in addition to mod_ssl

Modules – mod_ssl

- mod_ssl in **2.2**
  - SNI support (Server Name Indication)
    - SSL for name based VHosts
      - TLS extension: Client provides the server name already during the SSL handshake instead of only later in the Hostname HTTP header.
    - Increasing browser support
    - Did anyone already try this?

Modules – State Handling

- Cache infrastructure usable by modules: socache
  - „so“ means „Small Objects“
  - mod_socache_* has implementations „shmcb“, „dbm“, „memcache“ and „dc“ (distcache)
  - Used e.g. by mod_ssl (session cache) and mod_authn_socache (credential cache)
  - Caches are shared between all Apache processes of an instance

Modules – State Handling

- No longer using the scoreboard as hacked storage:
  - mod_slotmem_* implementations „plain“ and „shm“
    - plain: normal memory, not shared
    - shm: shared between processes
  - Difference to mod_socache: all slots of a slotmem have the same size
  - Currently used by
    - mod_proxy_balancer
    - Heartbeat

Modules – Caching

- mod_mem_cache was removed from 2.4
  - You should use mod_disk_cache instead, which has been renamed to mod_cache_disk!
  - mod_mem_cache is expected to be reimplemented using socache (mod_cache_socache)
- mod_cache_disk improved a lot
  - many fixes
  - many optimizations and better configurability

Modules – Hardening by using Connection Filters

- Security Improvements
  - mod_reqtimeout
    - Since 2.2.15
    - fine granular timeouts and minimum data rates
    - helps against Denial of Service by hanging or slow client connections (Slowloris etc.)
  - mod_ratelimit
    - simple bandwidth limitation per connection
  - mod_allowmethods
    - limits which HTTP methods are allowed

Modules – Watchdog

- mod_watchdog as a scheduler
  - modules register callbacks
  - mod_watchdog regularly calls the callbacks
  - One thread per child process, in addition usable as a singleton (one call per Apache instead of per process)
  - No guards against evil or long running callbacks

Modules – Heartbeat

- Heartbeat modules
  - mod_heartbeat: sends out the number of threads which are idle resp. busy via Multicast in regular intervals
    - uses mod_watchdog
    - should run in each node of a heartbeat controlled Apache farm
    - Configuration: address to which the data is being send

Modules – Heartbeat

- Heartbeat Receiver Nodes
  - mod_heartmonitor: listens to the data send by mod_heartbeat and maintains a list of farm nodes and current data
    - uses mod_watchdog
    - writes liste on disk (default) or to slotmem memory
  - mod_lbmethod_heartbeat: Provider für mod_proxy_balancer (load balancing)
    - balances requests according to load data distributed by mod_heartbeat
    - farm nodes do not need to be known/configured (auto-detect)

Modules – Varia

- Some modules we didn't mention
  - mod_buffer: input/output buffering
  - mod_request: reuse of request bodies
  - mod_reflector: send back the request body as the response
  - mod_session: cookie based session state handling
  - mod_auth_form: form based authentication
- Interesting separate or 3rd party modules
  - mod_fcgid, mod_ftp (httpd.apache.org)
  - mod_macro (http://www.coelho.net/mod_macro/)

Agenda

- Warm Up
- Version Overview
- MPMs
- Core: Logging, Expression Parser, Define
- Modules
- Community
- Discussion!

Community

- Documentation of 2.4 available unter http://httpd.apache.org/docs/2.4/
  - Always check the English documentation
    - Some translations are outdated – please contribute!
    - CAUTION: Your browser chooses the language
  - Some parts have improved a lot
    - mod_rewrite, mod_proxy, …
  - New comment feature: comments.apache.org
    - Implementation based on Lua
  - New syntax highlighting

Community

- Users list for questions: subscribe-users@httpd.apache.org
- Developer list (low-traffic): subscribe-dev@httpd.apache.org
- Current downloads should use the mirror system at http://httpd.apache.org/download.cgi
- Download archive with all releases available at http://archive.apache.org/dist/httpd
- Send bug reports to the users list first, escalate via the dev list or http://issues.apache.org/bugzilla/

Community

- Building 2.4 releases
  - Provide the needed dependencies (PCRE, OpenSSL), APR can be taken out of the 2.4.3-deps-tar.bz2 download
  - Run configure, make
- Building a snapshot
  - Provide the auto tools (autoconf/automake/libtool)
  - Check out http://svn.apache.org/repos/asf/httpd/httpd/branches/2.4.x/
  - Run „./buildconf --with-apr=[dir] and –with-apr-util=[dir]“
  - Run configure, make

Community

- Feedback welcome!
- Contributions even more!
  - Sometimes we (the project) are a bit slow in reacting :(
  - But we are usually not unfriendly ;)

Agenda

- Warm Up
- Version Overview
- MPMs
- Core: Logging, Expression Parser, Define
- Modules
- Community
- Discussion!

Discussion

- Any questions?