Hacking Apache Web Server

CONFidence – 13/05/2006

Błażej Miga, PCSS
blazej.miga@man.poznan.pl

Agenda

- Wprowadzenie
- Kilka słów o Apache
- Model działania serwera
- Moduły, uchwyty, filtry
- Prefork MPM
- Apache API
- Apache Analyzer
- Coś słodkiego…

ZB PCSS

- Praca operacyjna w ramach ogólnopolskiej szerokopasmowej sieci PIONIER oraz zasobów Centrum
- Praca badawczo – naukowa w ramach projektów krajowych oraz europejskich (CLUSTERIX, EGEE, UNIZETO, iTVP)
- Usługi komercyjne (audyty, testy penetracyjne, analizy kodów źródłowych i binarnych, szkolenia, etc...)
- Praca „po godzinach” (komunikatory, analizatory kodów źródłowych)

http://security.psnc.pl/

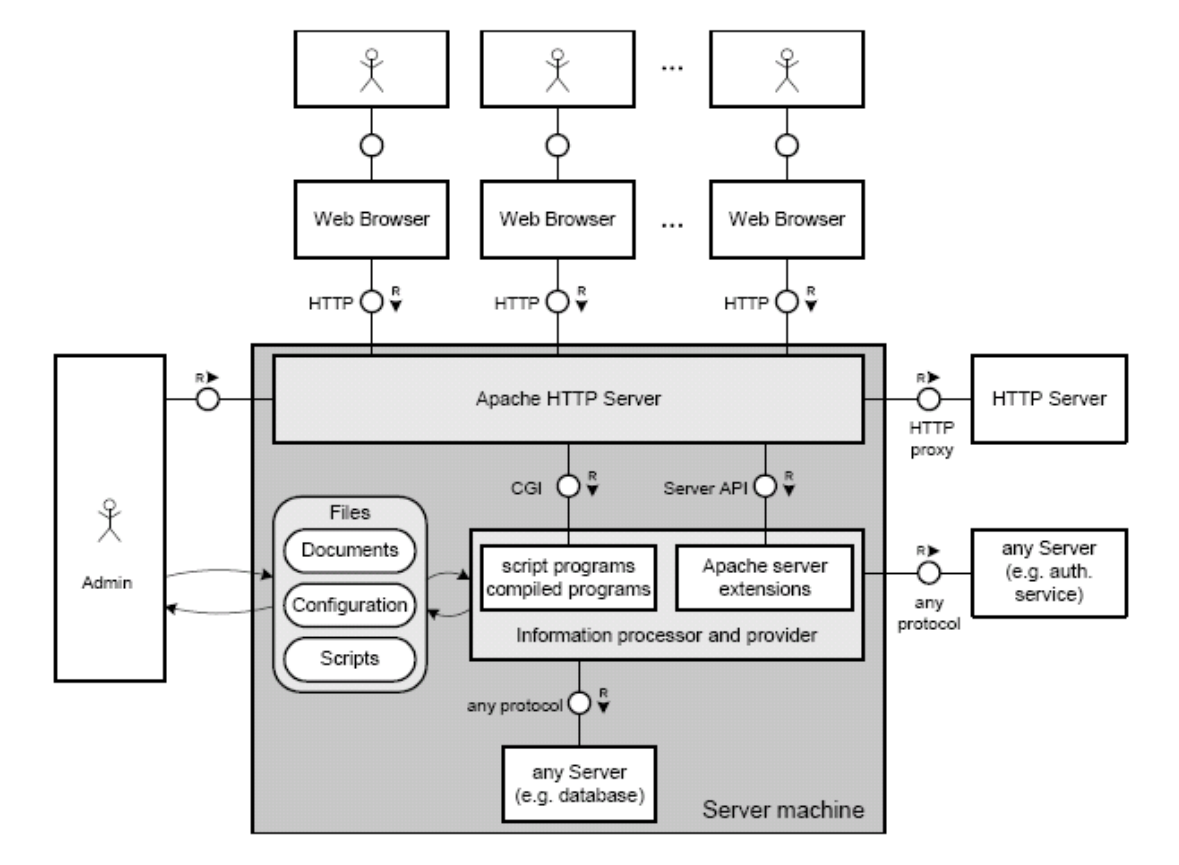
Historia i dzień dzisiejszy

- 1994 r. Rob McCool (National Center for Supercomputing Applications) NCSA HTTPd
- 1995 r. NCSA HTTPd v 1.3 - „a patchy” server – Apache
- Kwiecień 2006 - 68% rynku
- Aktualne wersje (2.2.2, 2.0.58 1.3.35)

Więcej informacji...

- Oficjalna strona internetowa
  - http://www.apache.org
- Listy mailingowe
  - announce@httpd.apache.org
  - dev@httpd.apache.org
- Apache Modeling Project
  - http://f-m-c.org/projects/apache/

Model działania serwera



Źródło: **„The Apache Modeling Project”** http://apache.hpi.uni-potsdam.de/

Moduły

- umożliwiają dodawanie nowych funkcjonalności do serwera
- kompilowane przy pomocy programu apxs
- wkompilowane w program lub w postaci bibliotek so
- wszystkie procesy Apache’a posiadają identyczny kod wykonywalny

Moduły

- moduły dynamiczne są wczytywane podczas startu serwer
- dostęp do struktur request_rec, server_rec, connection_rec
- rejestrują handlery dla standardowych hook lub innych modułów
- modules.apache.org – Baza modułów (426 - 05.2006)

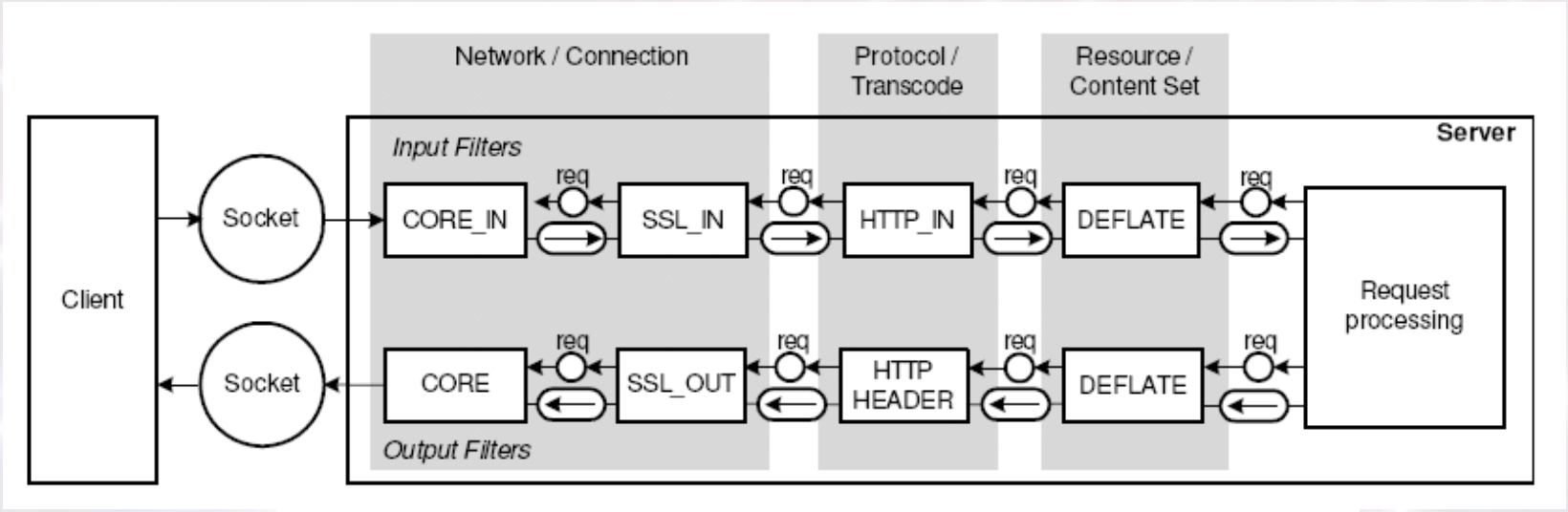
Handlers

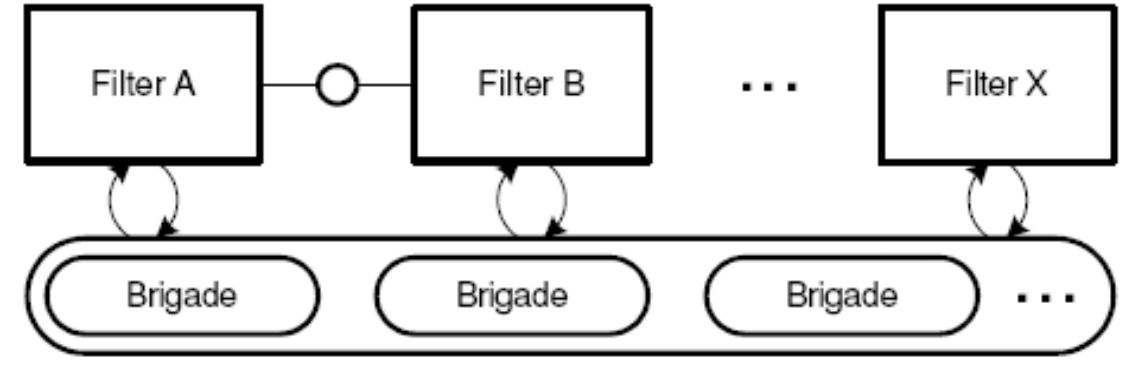
- Handlers for hooks
- Handlers for dealing with configuration directives
- Filters
- Optional functions
- Content handlers

Filtry

- Umożliwiają zmianę zawartości danych odbieranych (filtry wejściowe) i wysyłanych (filtry wyjściowe) do przeglądarki klienta
- Filtry umożliwiają zmianę danych na poziomie sieciowym, protokołu lub przesyłanych danych HTTP.

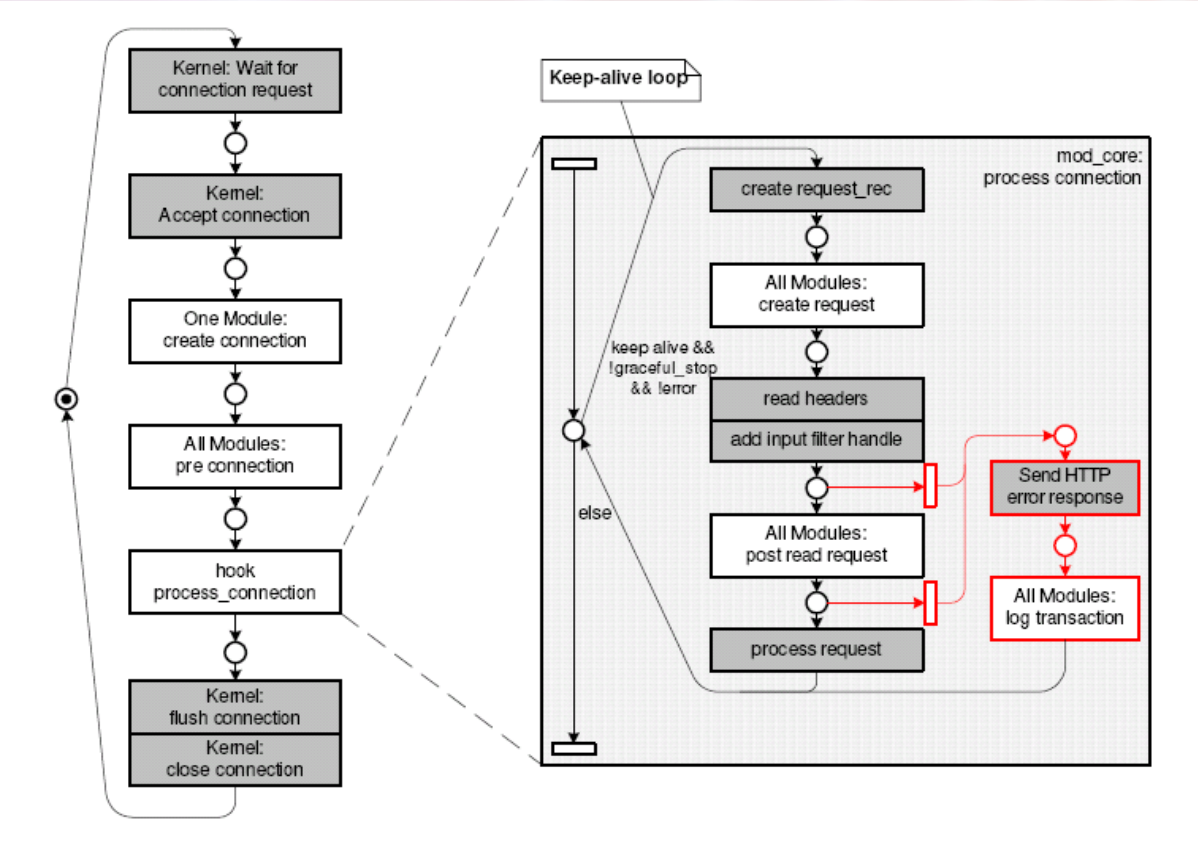
Filtry





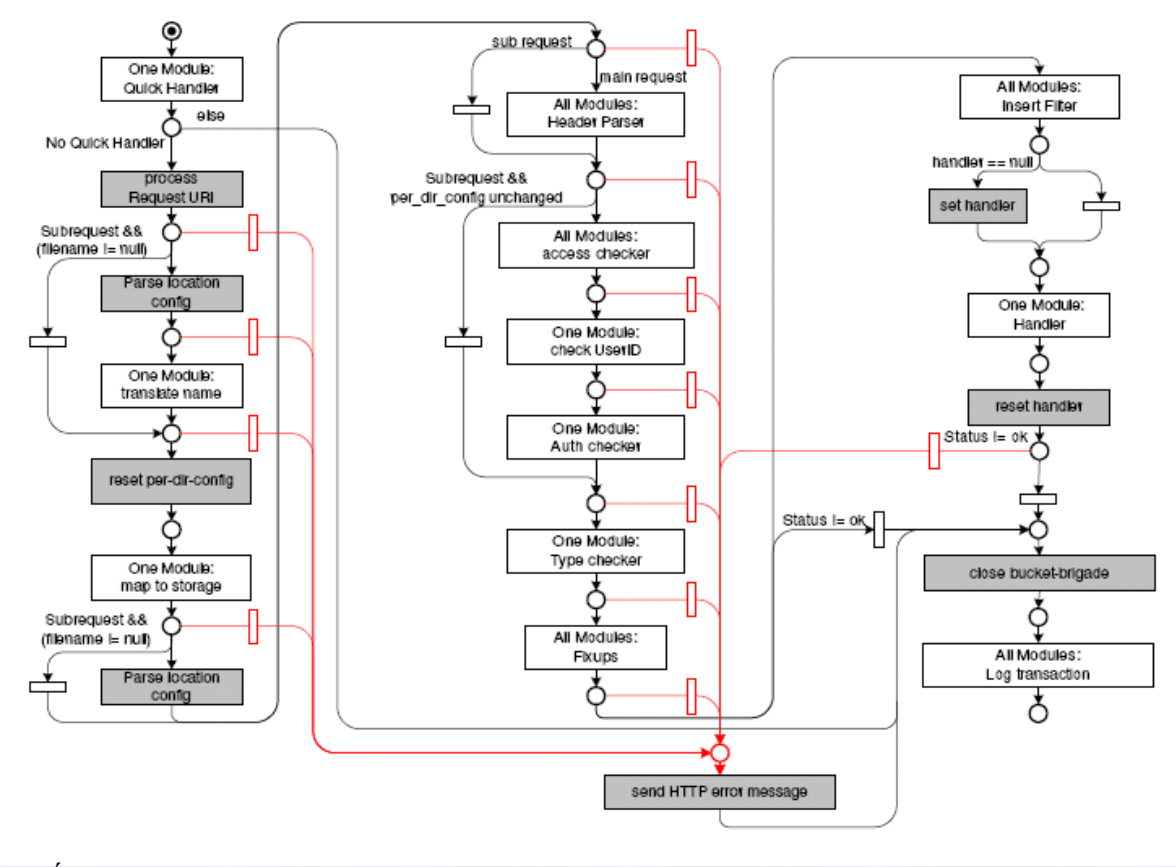
Źródło: **„The Apache Modeling Project”** http://apache.hpi.uni-potsdam.de/

Obsługa żądań HTTP



Źródło: **„The Apache Modeling Project”** http://apache.hpi.uni-potsdam.de/

Obsługa żądań HTTP

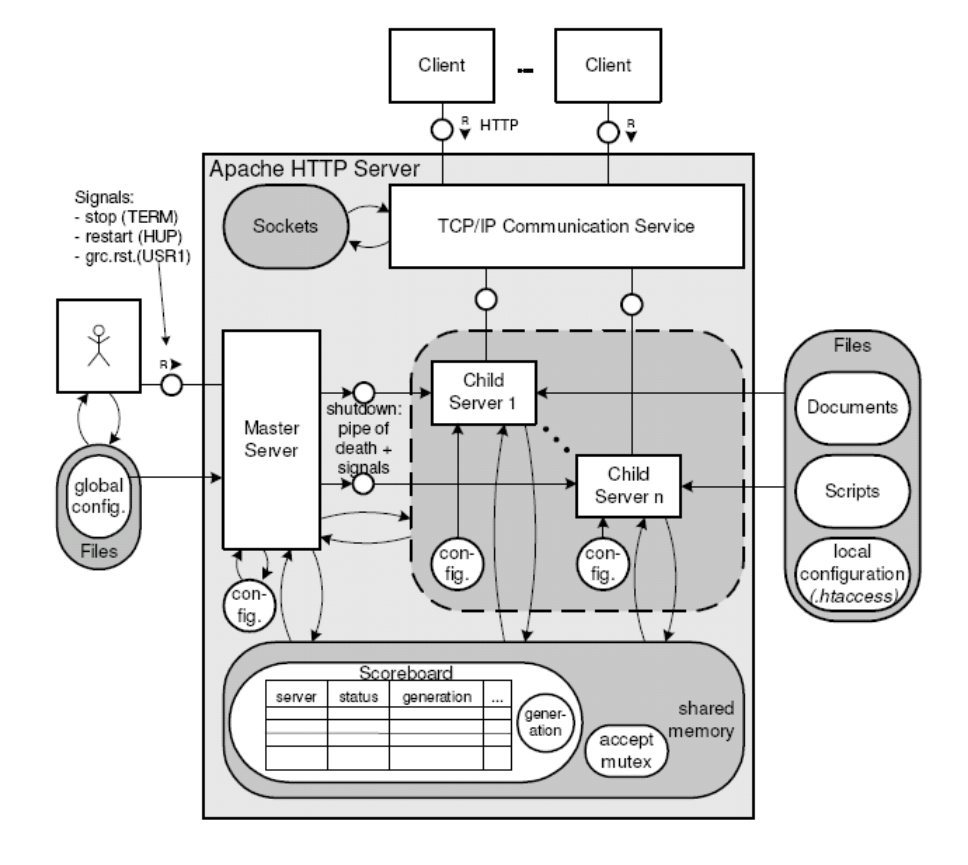


Źródło: **„The Apache Modeling Project”** http://apache.hpi.uni-potsdam.de/

Prefork MPM

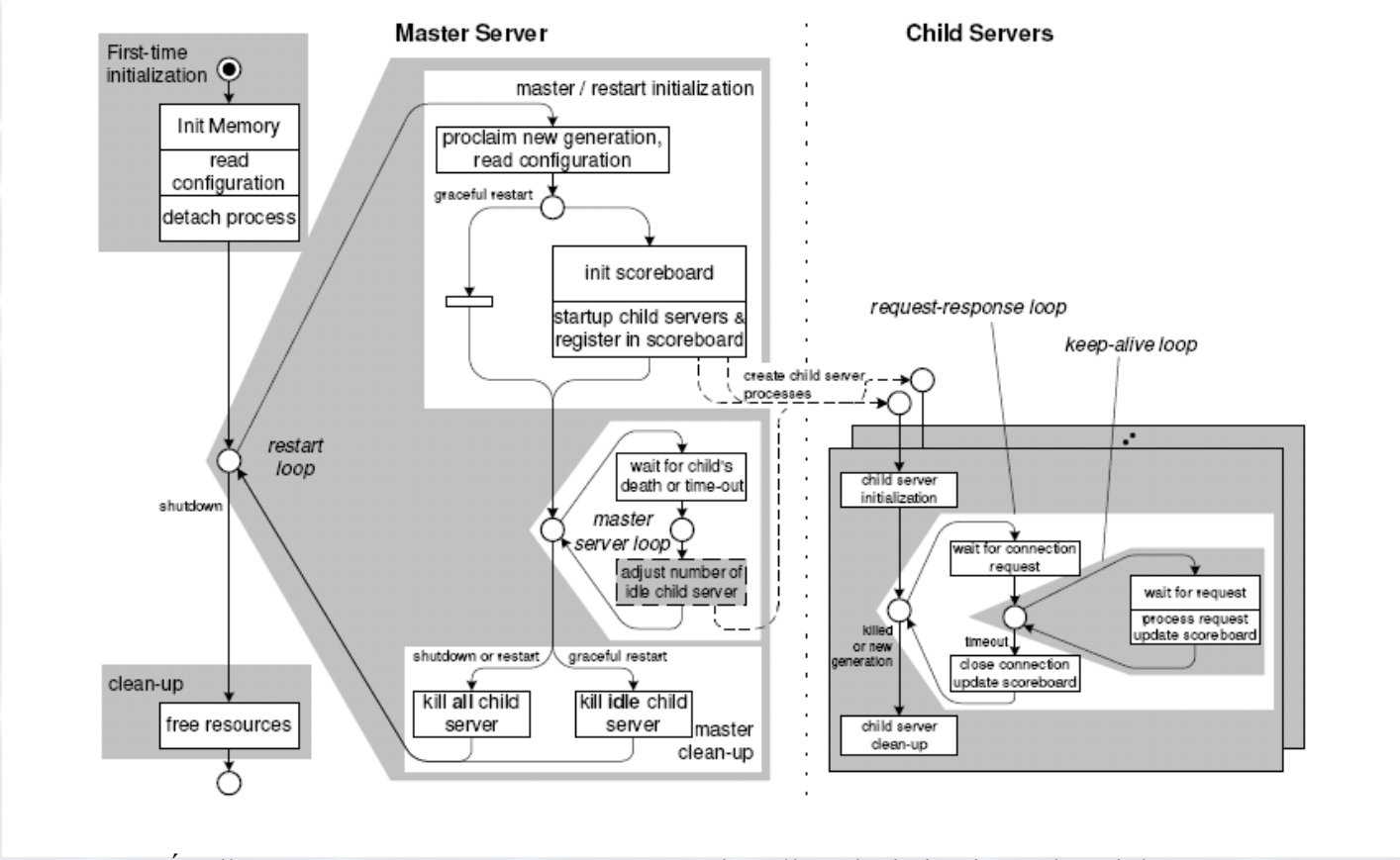
- MPM – Multi-Processing Modules
- Pierwsza multitaskowa architektura Apache'a
- Domyślny MPM dla UNIX
- Jeden proces server – zarządca – ustala liczbę procesów - dzieci zajmujące się żądaniami HTTP

Prefork MPM



Źródło: **„The Apache Modeling Project”** http://apache.hpi.uni-potsdam.de/

Prefork MPM



Źródło: **„The Apache Modeling Project”** http://apache.hpi.uni-potsdam.de/

Apache API

- Zarządzanie pamięcią
- Multitasking
- Operacje na tablicach i stringach
- Operacje sieciowe
- Funkcje logujące
- Identyfikacja, autoryzacja, uwierzytelnianie

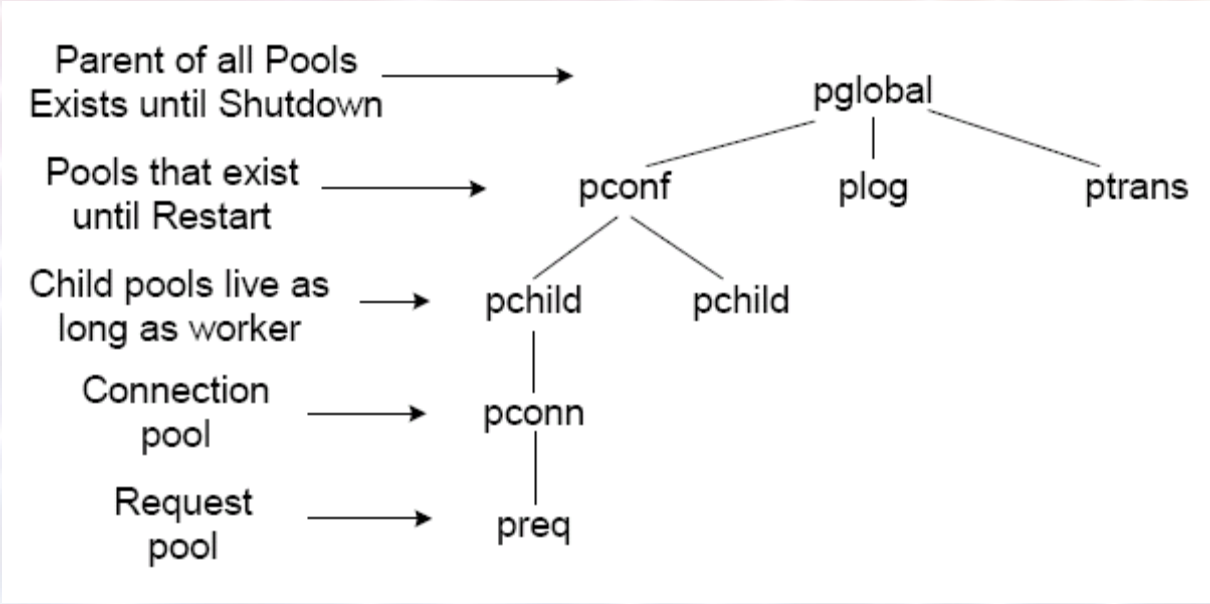
Apache Portable Runtime

- Dodano od wersji 2.0
  - File I/O + Pipes
  - Zaawansowane zarządzanie pamięcią
  - Wątki
  - Dynamiczne linkowanie
  - Funkcje obsługi czasu

Pools

- służą do zarządzanie pamięcią
- tworzone są w pamięci heap
- wbudowane pule pglobal, pconf, plog, ptrans, pchild, pconn, preq
- każdy typ posiada różny czas życia
- tworzą strukturę drzewa

Pools


Parent of all Pools
Exists until Shutdown
Pools that exist
until Restart
Child pools live as
long as worker
Connection
pool
Request
pool
pglobal
pconf
plog
ptrans
pchild
pchild
pconn
preq

Najważniejsze struktury

- request_rec
  - najważniejsza struktura
  - zawiera informacje o żądaniu HTTP
  - zawiera informacje o serwerze do którego przyszło żądanie
  - zawiera informacje o połączeniu do którego należy
  - zawiera wskaźnik do puli danych żądania i odpowiedzi

Request_rec

```
struct request_rec {
apr_pool_t *pool;
conn_rec *connection;
server_rec *server;
request_rec *next;
request_rec *prev;
request_rec *main;
char *the_request;
int assbackwards;
int proxyreq;
int header_only;
char *protocol;
int proto_num;
const char *hostname;
apr_time_t request_time;
const char *status_line;
int status;
const char *method;
int method_number;
apr_int64_t allowed;
ap_method_list_t *allowed_methods;
apr_off_t sent_bodyct;
apr_off_t bytes_sent;
apr_time_t mtime;
const char *range;
apr_off_t clength;
apr_off_t remaining;
apr_off_t read_length;
int read_body;
int read_chunked;
apr_table_t *headers_in;
apr_table_t *headers_out;
apr_table_t *err_headers_out;
apr_table_t *subprocess_env;
apr_table_t *notes;
const char *content_type;
const char *handler;
const char *content_encoding;
apr_array_header_t *content_languages;
char *vlist_validator;
char *user;
char *ap_auth_type;
char *unparsed_uri;
char *uri;
char *filename;
char *canonical_filename;
char *path_info;
char *args;
apr_finfo_t finfo;
apr_uri_t parsed_uri;
struct ap_conf_vector_t *per_dir_config;
struct ap_conf_vector_t *request_config;
const struct htaccess_result *htaccess;
struct ap_filter_t *output_filters;
struct ap_filter_t *input_filters;
struct ap_filter_t *proto_output_filters;
struct ap_filter_t *proto_input_filters;
};
```

Najważniejsze struktury

- server_rec
  - zawiera informacje o serwerze obsługującym żądanie
  - informacja o konfiguracji serwera i wirtualnych hostach
  - informacja o nazwie serwera, portach na których nasłuchuje
  - konfiguracje specyficzna do danego serwera

Server_rec

```
struct server_rec {
process_rec *process;
server_rec *next;
const char *defn_name;
unsigned defn_line_number;
char *server_admin;
char *server_hostname;
apr_port_t port;
char *error_fname;
apr_file_t *error_log;
int loglevel;
int is_virtual;
struct ap_conf_vector_t *module_config;
struct ap_conf_vector_t *lookup_defaults;
server_addr_rec *addrs;
apr_interval_time_t timeout;
apr_interval_time_t keep_alive_timeout;
int keep_alive_max;
int keep_alive;
const char *path;
int pathlen;
apr_array_header_t *names;
apr_array_header_t *wild_names;
int limit_req_line;
int limit_req_fieldsize;
int limit_req_fields;
```

- };

Najważniejsze struktury

- conn_rec
  - HTTP 1.1 umożliwia realizowanie wielu żądań poprzez jedno połączenie
  - posiada informacje o aktualnym połączeniu
  - zawiera informacje o serwerze do którego zostało zrealizowane połączenie

Conn_rec

```
struct conn_rec {
apr_pool_t *pool;
server_rec *base_server;
void *vhost_lookup_data;
apr_sockaddr_t *local_addr;
apr_sockaddr_t *remote_addr;
char *remote_ip;
char *remote_host;
char *remote_logname;
unsigned aborted:1;
ap_conn_keepalive_e keepalive;
signed int double_reverse:2;
int keepalives;
char *local_ip;
char *local_host;
long id;
struct ap_conf_vector_t *conn_config;
struct ap_filter_t *input_filters;
struct ap_filter_t *output_filters;
void *sbh;
struct apr_bucket_alloc_t *bucket_alloc;
conn_state_t *cs;
int data_in_input_filters;
};
```

Apache Analyzer

- Prezentacja działania programu
  - Informacje o systemie
  - Wyszukiwanie konfiguracji serwera
  - Odnajdywanie najważniejszych struktur serwera:
    - pools, moduły, hooki, filtry,
    - struktury server_info i request_info
    - scoreboard
- Niespodzianka!

Ochrona systemu

- Blokada niebezpiecznych funkcji PHP (dl, pcntl_exec, posix_kill, system, exec, passthru, popen, proc_open, shell_exec, mail …)
- Wyłączenie modułu mod_perl
- Stosowanie virtualhost tylko dla zaufanych domen

Dane kontaktowe

blazej.miga@man.poznan.pl
http://www.man.poznan.pl
http://security.psnc.pl

Pytania?

Dziękuję za uwagę!