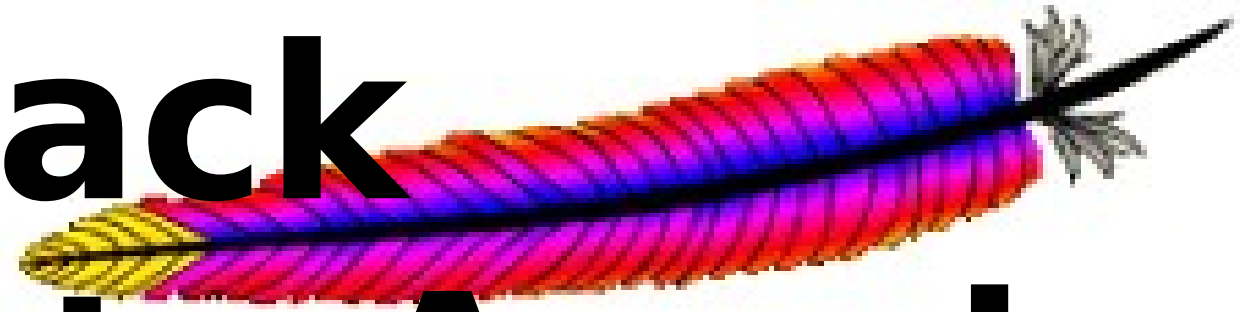

H2HC
APRESENTA


nerv
LABS

# Hack into Apache



**Objetivo: adentrar nas estruturas internas do servidor web apache, tal como exemplificar casos de módulos inseguros e técnica úteis na exploração de falhas de segurança.**

# Tópicos

- Falhas de 2007...
- Apache em modo verboso
- Estruturas Internas
- mod_vuln.c

# Falhas conhecidas

Apenas em 2007

# mod_proxy, mod_cache...

Falhas patcheadas nesta última versão (2.2.6):

moderate: mod_proxy crash
moderate: mod_status cross-site scripting
moderate: Signals to arbitrary processes (local)
moderate: mod_cache information leak
moderate: mod_cache proxy DoS

# Modo Verboso.

Prolixo e Gongórico.

# Compilando...

**http://httpd.apache.org/docs/2.2/programs/configure.html**

```
AP_VERSION=2.2.6
AP_PATH=/usr/local/apache

cd httpd-$AP_VERSION
./configure \
        --prefix=$AP_PATH \
        --enable-maintainer-mode \
        --enable-exception-hook \
        --enable-mods-shared=most
make && make install
```

# --enable-maintainer-mode

- Habilita modo de depuração, a vida fica mais bela no *gdb*!

- Macros interessantes do *.gdbinit:* *dump_brigade* & *dump_bucket*
- Bom lugares para *breakpoints:* *ap_pass_brigade()* & *core_output_filter()*

# --enable-exception-hook

- Disponível apatir da versão 2.0.49
- Habilita *hook* para *fatal exceptions*

- **mod_backtrace**: exibe *backtrace* dos *frames* apatir do momento do crash.

http://people.apache.org/~trawick/mod_backtrace.c

- **mod_whatkilledus**: dump da requisição no momento do crash.

http://people.apache.org/~trawick/mod_backtrace.c

# mod_backtrace && mod_whatkilledus

```
vs_apache:/usr/local/apache/logs# kill -SIGSEGV `pgrep http| tail -1` | tail -n 19 error_log
[Sun Nov  4 15:13:24 2007] pid 11263 mod_backtrace backtrace for sig 11 (thread "pid" 11263)
[Sun Nov  4 15:13:24 2007] pid 11263 mod_backtrace main() is at 80617a0
/usr/local/apache-2.2.6-debug/modules/mod_backtrace.so[0xb7d2bb4e]
/usr/local/apache-2.2.6-debug/bin/httpd(ap_run_fatal_exception+0x39)[0x807ca19]
/usr/local/apache-2.2.6-debug/bin/httpd[0x807cdf0]
[0xffffe420]
/usr/local/apache-2.2.6-debug/bin/httpd(unixd_accept+0x2c)[0x808812c]
/usr/local/apache-2.2.6-debug/bin/httpd[0x8086939]
/usr/local/apache-2.2.6-debug/bin/httpd[0x8086cfa]
/usr/local/apache-2.2.6-debug/bin/httpd(ap_mpm_run+0x85e)[0x808762e]
/usr/local/apache-2.2.6-debug/bin/httpd(main+0x85f)[0x8061fff]
/lib/tls/libc.so.6(__libc_start_main+0xc8)[0xb7dffea8]
/usr/local/apache-2.2.6-debug/bin/httpd(apr_bucket_mmap_make+0x6d)[0x8061201]
[Sun Nov  4 15:13:24 2007] pid 11263 mod_backtrace end of backtrace
[Sun Nov  4 15:13:24 2007] pid 11263 mod_whatkilledus sig 11 crash
[Sun Nov  4 15:13:24 2007] pid 11263 mod_whatkilledus no active connection at crash
[Sun Nov  4 15:13:24 2007] pid 11263 mod_whatkilledus no request active at crash
[Sun Nov  4 15:13:24 2007] pid 11263 mod_whatkilledus end of report
[Sun Nov 04 15:13:24 2007] [notice] child pid 11263 exit signal Segmentation fault (11)
vp_apache:/usr/local/apache/logs#
```

# mod_dumpio

- Disponível apatir da versão 2.1.3
- Registra **todos** os dados logo após o *decode*/*encode* do mod_ssl, tanto na entrada como na saída!

# mod_dumpio

```
[debug] mod_dumpio.c(113): mod_dumpio: dumpio_in [getline-blocking] 0 readbytes
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_in (data-HEAP): 15 bytes
[debug] mod_dumpio.c(74): mod_dumpio:  dumpio_in (data-HEAP): GET / HTTP/1.0\n
[debug] mod_dumpio.c(113): mod_dumpio: dumpio_in [getline-blocking] 0 readbytes
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_in (data-HEAP): 11 bytes
[debug] mod_dumpio.c(74): mod_dumpio:  dumpio_in (data-HEAP): Host: Teste\n
[debug] mod_dumpio.c(113): mod_dumpio: dumpio_in [getline-blocking] 0 readbytes
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_in (data-HEAP): 1 bytes
[debug] mod_dumpio.c(74): mod_dumpio:  dumpio_in (data-HEAP): \n
[debug] mod_dumpio.c(142): mod_dumpio: dumpio_out
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_out (data-HEAP): 175 bytes
[debug] mod_dumpio.c(74): mod_dumpio:  dumpio_out (data-HEAP): HTTP/1.1 200 OK\r\nDate: Sun,
04 Nov 2007 16:53:44 GMT\r\nServer: Apache/2.2.6 (Unix) DAV/2\r\nContent-Length:
535\r\nConnection: close\r\nContent-Type: text/html;charset=ISO-8859-1\r\n\r\n
[debug] mod_dumpio.c(142): mod_dumpio: dumpio_out
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_out (data-HEAP): 535 bytes
[debug] mod_dumpio.c(74): mod_dumpio:  dumpio_out (data-HEAP): <!DOCTYPE HTML PUBLIC "-
//W3C//DTD HTML 3.2 Final//EN">\n<html>\n <head>\n  <title>Index of /</title>\n </head>\n
<body>\n<h1>Index of /</h1>\n<ul><li><a href="apache_pb.gif"> apache_pb.gif</a></li>\n<li><a
href="apache_pb.png"> apache_pb.png</a></li>\n<li><a href="apache_pb22.gif">
apache_pb22.gif</a></li>\n<li><a href="apache_pb22.png"> apache_pb22.png</a></li>\n<li><a
href="apache_pb22_ani.gif"> apache_pb22_ani.gif</a></li>\n<li><a href="helloworld/">
helloworld/</a></li>\n<li><a href="index.html"> index.html</a></li>\n</ul>\n</body></html>\n
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_out (metadata-EOS): 0 bytes
[debug] mod_dumpio.c(142): mod_dumpio: dumpio_out
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_out (metadata-FLUSH): 0 bytes
[debug] mod_dumpio.c(142): mod_dumpio: dumpio_out
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_out (metadata-FLUSH): 0 bytes
[debug] mod_dumpio.c(55): mod_dumpio:  dumpio_out (metadata-EOC): 0 bytes
```

# Depurando *Filters*

- **mod_filter**: realiza *dumps* dos *buckets* antes e após de um dado *Filter.*

No httpd.conf do apache:

"FilterTracer *filter-name level"*

Exemplo: "FilterTracer myfilter debug"

- **mod_diagonistic**: *third-party* module que faz a mesma coisa...

# Estruturas Internas

Use the source, luke!

# Tópicos: das entranhas...

- O Código
- *Modules* e *Hooks*
- *Buckets*, *Brigades* e *Filters*
- *APR, MPMs*

# Sobre o código

- 1.x: o início
- 2.0.x: a portabilidade (apr)
- 2.2.x: *code refactor* dos módulos, configuração modularizada, mod_proxy_balance...
- +300K linhas de código na 2.2.6

# Mais sobre o código

- server/ : core
- include/ : headers do core
- srclib/apr/ : biblioteca apr
- srclib/apr-util/ : utilitários com apr
- support/ : infraestrutura de suporte
- modules/ : código de módulos
- docs/ : documentação

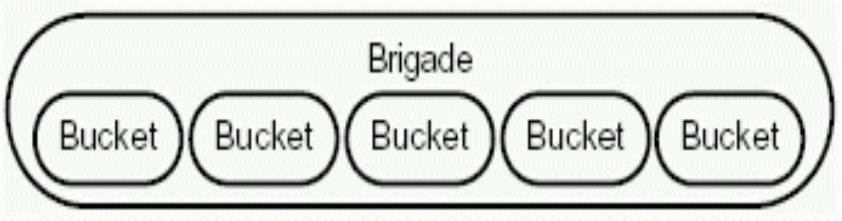
# *Buckets, Brigades* e *Filters*

- *Buckets*: abstração da unidade mínima de dados.
- *Brigades*: cadeias de *buckets, é o* controle de uma lista encadeada.
- *Filters*: entidades que acessam as brigades buscandos dados disponíveis nos buckets.

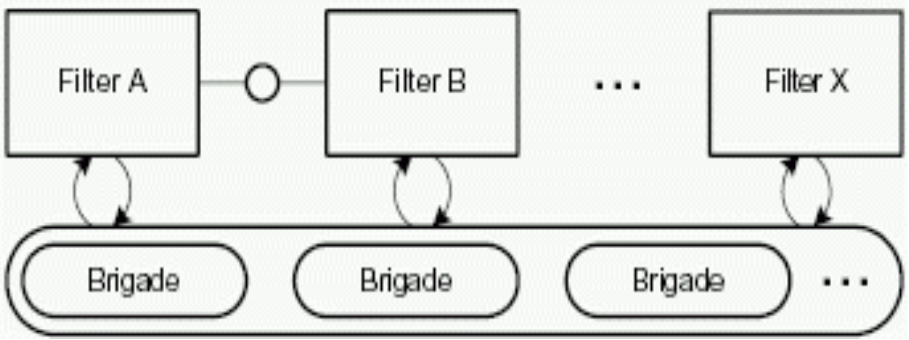
# *Buckets, Brigades* e *Filters*

- *Buckets*: abstração da unidade mínima de dados.
- *Brigades*: cadeias de *buckets, é o* controle de uma lista encadeada.
- *Filters*: entidades que acessam as brigades buscandos dados disponíveis nos buckets.

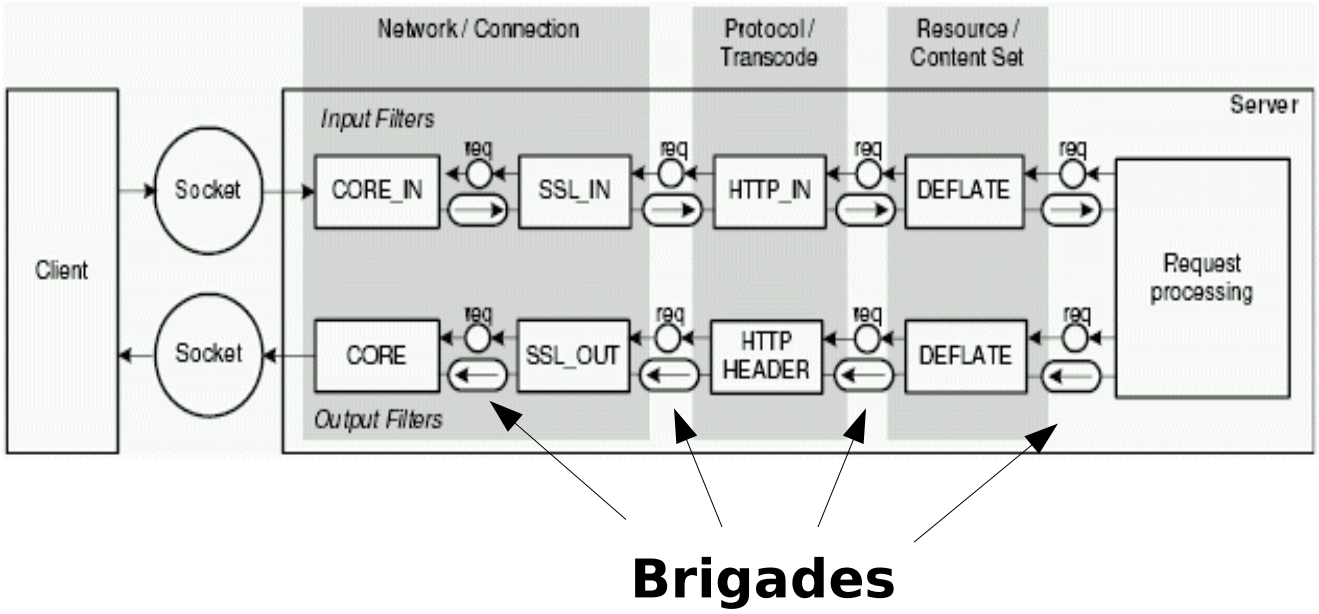
## Bucket && Brigades




## Filters -> Brigades




## Filters:
1. Conection
2. Protocol
3. Content

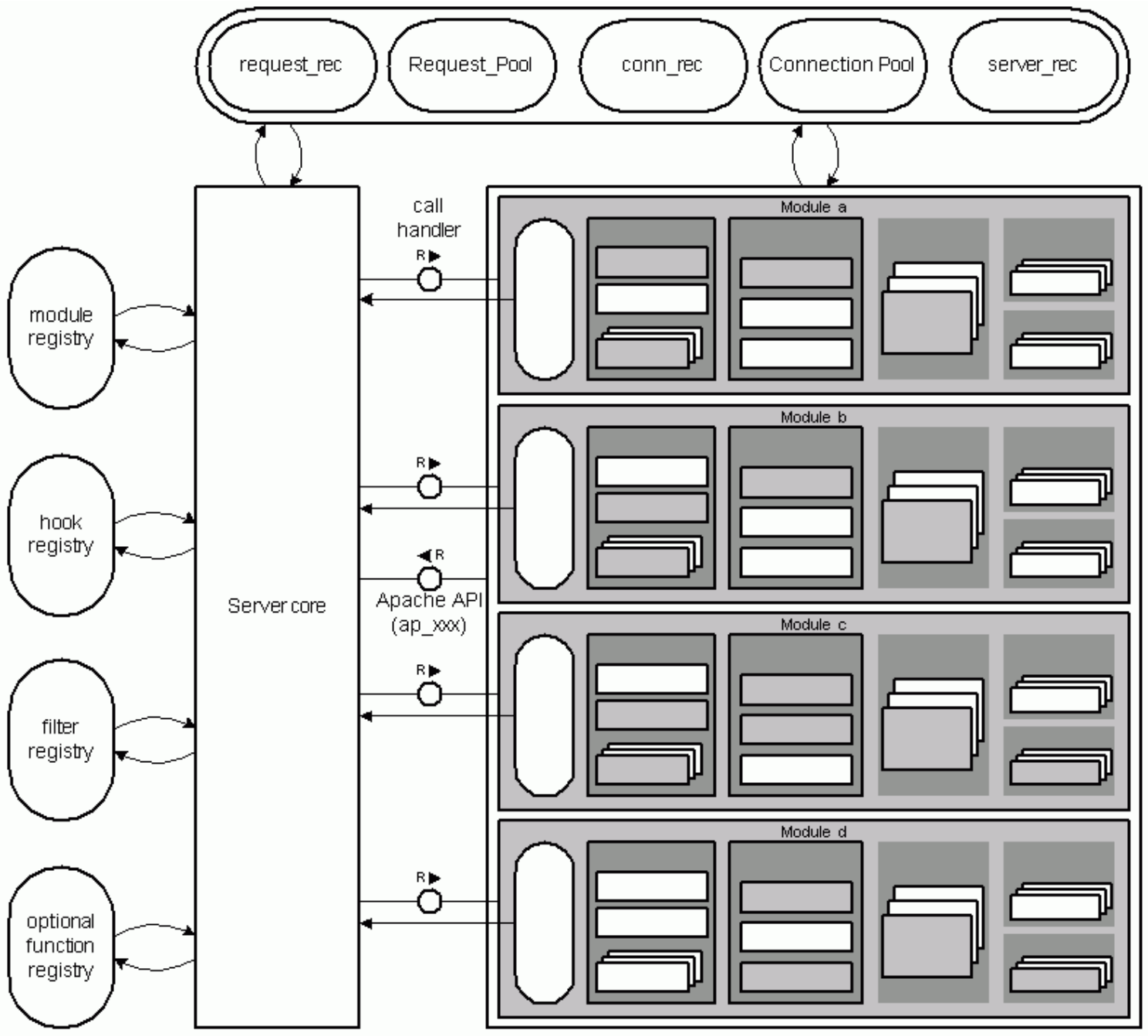


# *Modules* e *Hooks*

- Module: extendem as funcionalidades do *core* do apache. Exemplos são a mod_ssl integrando a OpenSSL e o mod_perl adicionando suporte a Perl.

- Hooks: são *callbacks* disponibilizadas pelo *core*. Apartir dessas temos acesso a pontos específicos durante um dada requisição.

# Modules

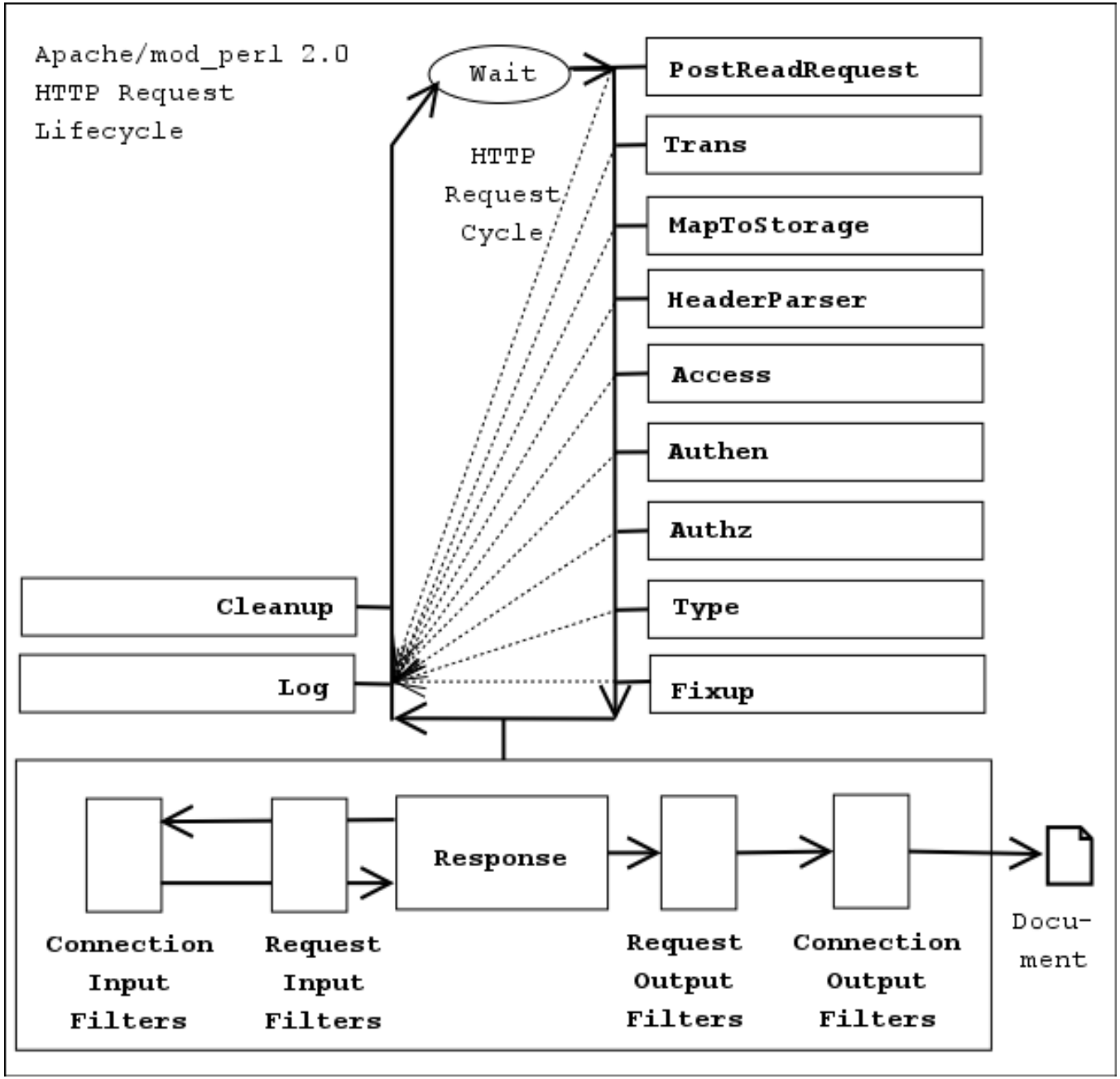

request_rec
Request_Pool
conn_rec
Connection Pool
server_rec
call handler
R
module registry
hook registry
filter registry
optional function registry
Server core
Apache API (ap_xxx)
Module a
Module b
Module c
Module d

# HTTP: Ciclo de vida

O caminho de uma requisição...

- *Input Filters*
- *Apache Hooks*
- *Output Filters*

# Ciclo de vida uma requisição



# APR: Apache Portable Runtime

Funcionalidades *cross-plataform* via APR:

- Allocação de Memória
- E/S de arquivos
- *Sockets* de rede
- ...

Outros projetos fazem uso (e.g. svn)

# MPM (Multi-processing modules)

MPMs: *dispatcher* das *request* para os processos e as threads. No UNIX:

- ***prefork***: *non-threaded*. (*default*)
- ***workers***: *multi-process* e *multi-threads implementation*.

# Apache Pools

- *Pools,* tal como um *garbage collector*, se responsabilizam de liberarar a memória.

```
type* var = malloc(sizeof(type));
type* var = apr_palloc(pool, sizeof(type));
```

- Exemplo: quando uma *request* acaba toda memória de um *pool* é liberada.
- Tipos: *request pool*, *process pool*, *connection pool*, *configuration pool*...

# mod_vuln.c

Exemplificando...

# mod_vuln: qual é a idéia?

Criar um “mod_vuln”.

Um módulo para “extensão” de funcionalidades do servidor apache com uma falha clássica de segurança (no caso buffer overfllow) e promover uma exploração desta falha nesse ambiente.

# mod_vuln: tem funcionalidade?

Acessa **`request_req->headers_in->{HOST}`**
e via **`ap_log_error()`** faz um registro nos logs do apache (ver **`logs/error_log`**).

# Como integrar? (**apxs**)

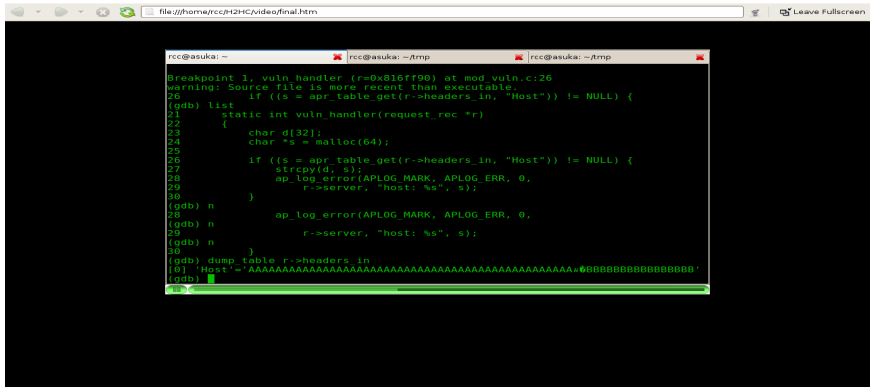
```
# pwd
/usr/local/apache
# ls mod_vuln/
mod_vuln.c
# bin/apxs -ci mod_vuln/mod_vuln.c
(....)
# cat conf/httpd.conf | grep mod_vuln
LoadModule vuln_module        modules/mod_vuln.so
# ls modules/ | grep mod_vuln
mod_vuln.so
```

```c
/* Apache's Fixup Vuln Handler Module  */

#include "httpd.h"
#include "http_config.h"
#include "http_core.h"
#include "http_log.h"
#include "http_protocol.h"
#include "ap_compat.h"

static void register_hooks(apr_pool_t *p);
static int vuln_handler(request_rec *r);

static void register_hooks(apr_pool_t *p)  /* use ap_hook_fixup() to register */
{
    ap_hook_fixups(vuln_handler, NULL, NULL, APR_HOOK_MIDDLE);
}

static int vuln_handler(request_rec *r)    /* do whatever you want */
{
    char d[32];
    char *s = malloc(64);

    if ((s = apr_table_get(r->headers_in, "Host")) != NULL) {
        strcpy(d, s);
        ap_log_error(APLOG_MARK, APLOG_ERR, 0,
            r->server, "host: %s", s);
    }
    return OK;
}

module AP_MODULE_DECLARE_DATA vuln_module =
{
    STANDARD20_MODULE_STUFF,
    NULL,                       /* create per-directory config structure           */
    NULL,                       /* merge per-directory config structures           */
    NULL,                       /* create per-server config structure              */
    NULL,                       /* merge per-server config structures              */
    NULL,                       /* command apr_table_t                             */
    register_hooks              /* register hooks                                  */
};
```

# mod_backtrace && mod_whatkilledus

```
[Sun Nov  4 15:13:24 2007] pid 11263 mod_backtrace backtrace for sig 11 (thread "pid" 11263)
[Tue Nov  6 08:46:26 2007] pid 8963 mod_backtrace backtrace for sig 11 (thread "pid" 8963)
[Tue Nov  6 08:46:26 2007] pid 8963 mod_backtrace main() is at 80617a0
/usr/local/apache-2.2.6-debug/modules/mod_backtrace.so[0xb7ce3b4e]
/usr/local/apache-2.2.6-debug/bin/httpd(ap_run_fatal_exception+0x39)[0x807ca19]
/usr/local/apache-2.2.6-debug/bin/httpd[0x807cdf0]
[0xffffe420]
[Tue Nov  6 08:46:26 2007] pid 8963 mod_backtrace end of backtrace
[Tue Nov  6 08:46:26 2007] pid 8963 mod_whatkilledus sig 11 crash
[Tue Nov  6 08:46:26 2007] pid 8963 mod_whatkilledus active connection: 192.168.1.1:5153041-
>192.168.1.1:80 (conn_rec 8169e98)
[Tue Nov  6 08:46:26 2007] pid 8963 mod_whatkilledus active request (request_rec 816ff60):
GET / HTTP/1.0|Host:AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA%ef%be%ad%de[Tue Nov  6
08:46:26 2007] pid 8963 mod_whatkilledus end of report
```

# Abra o video.


file:///home/rcc/H2HC/video/final.htm
Leave Fullscreen
rcc@asuka: ~
rcc@asuka: ~/tmp
rcc@asuka: ~/tmp
Breakpoint 1, vuln_handler (r=0x816ff90) at mod_vuln.c:26
warning: Source file is more recent than executable.
26 if ((s = apr_table_get(r->headers_in, "Host")) != NULL) {
(gdb) list
21 static int vuln_handler(request_rec *r)
22 {
23 char d[32];
24 char *s = malloc(64);
25
26 if ((s = apr_table_get(r->headers_in, "Host")) != NULL) {
27 strcpy(d, s);
28 ap_log_error(APLOG_MARK, APLOG_ERR, 0,
29 r->server, "host: %s", s);
30 }
(gdb) n
28 ap_log_error(APLOG_MARK, APLOG_ERR, 0,
(gdb) n
29 r->server, "host: %s", s);
(gdb) n
30 }
(gdb) dump table r->headers_in
[0] 'Host'~'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA»BBBBBBBBBBBBBBB'
(gdb)

# Nota sobre o sistema alvo

Todos os testes foram realizados em um sistema **Debian Etch** com kernel **2.6.21.6** com os patchs “**vserver**” e “**grsecurity**” com configurações defaults.

O apache com foi compilado dentro de uma jaula de um “vserver” com **gcc 4.1**, versão default do debian stable.

# Dúvidas

?

# Referências

http://www.apachetutor.org/dev
http://www.fmc-modeling.org/projects/apache
http://people.apache.org/~trawick/exception_hook.html
http://perl.apache.org/docs/2.0/user/handlers/http.html

**"There is only information and those that can invoked it."**

**- Phantasmal Phantasmagoria**