Y36AWS Teoretická zkouška 11.12.2008

Z STM Wiki

Důležité je, že ani při jedné z částí není možno mít žádné pomůcky. Teorie je z paměti a k praktické části máte akorát na localhostu dostupnou dokumentaci Apache, internet je odpojen. Celkově zkouška není tak těžká, na přípravu myslím bohatě stačí absolvovaná cvičení a přednášky dostupné na webu předmětu.

Teoretická část je vypisovací test asi o deseti otázkách typu:

Popište podrobně handshake SSLv3/TLSv1

1. dohoda o šifrách
2. vytvoření session
3. volitelně autentizace serveru a klienta

Konkrétně:

- klient pošle úvodní zprávu Hello a seznam podporovaných šifrovacích algoritmů
- server pošle certifikát obsahující veřejný klíč serveru, šifrovací algoritmus z klientského seznamu, který bude používat
- klient ověří certifikát pomocí root certifikátu CA
- pomocí náhodných čísel a šifry klient vygeneruje klíč pro symetrické šifry a zašifruje ji veřejným klíčem serveru
- server si klíč rozšifruje svým privátním klíčem a už vesele šifrovaně komunikují

viz obrázek

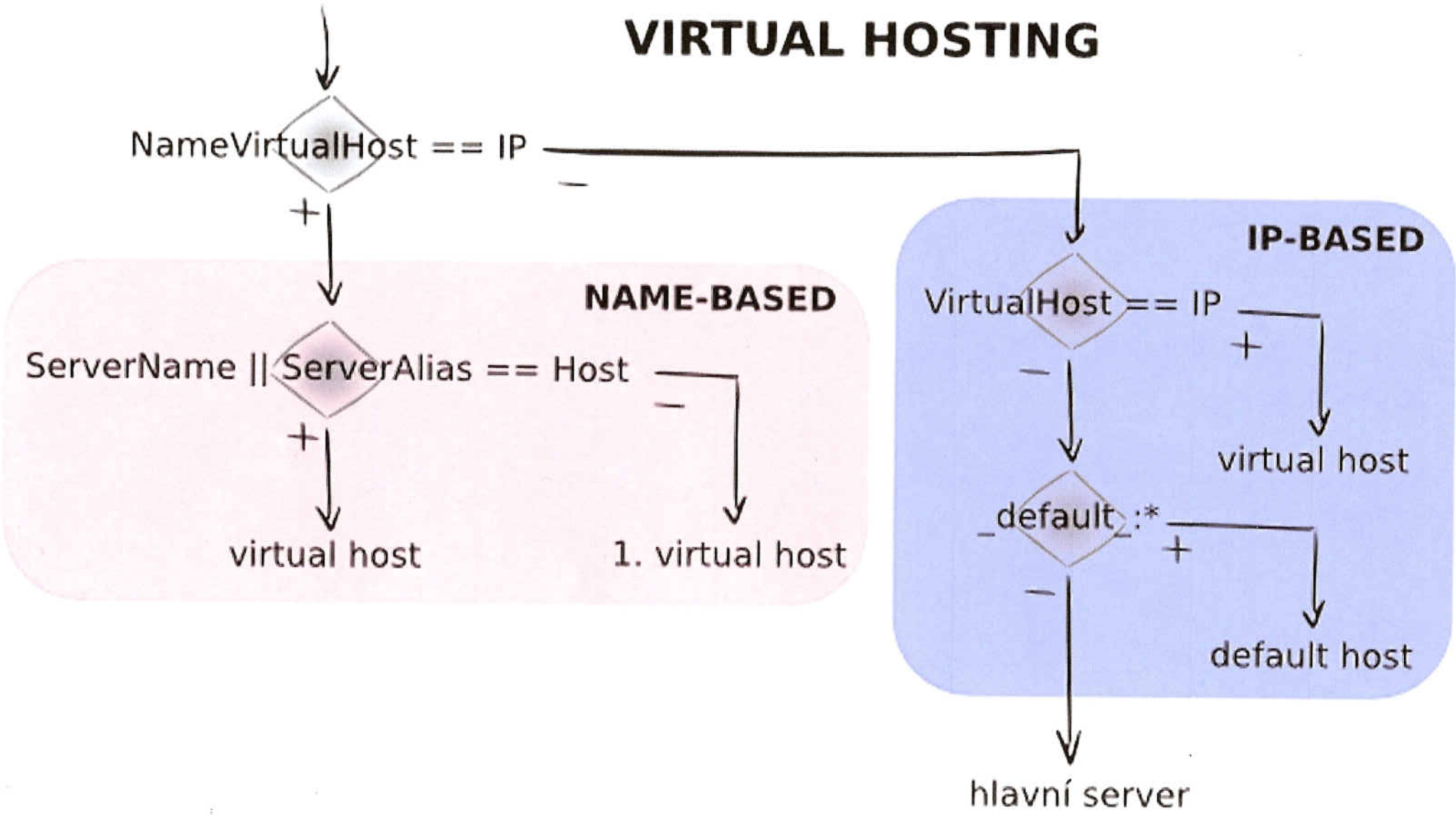
Napište alespoň tři body kontrolního algoritmu SUEXEC.

- cgi script nevlastní root a nikdo jiný krom uživatele do něj nemůže zapisovat
- nikdo jiný nemůže zapisovat ani do nadřazeného adresáře
- uživatel splňuje požadavake na mininální uid zvolené při kompilaci
- cgi script nesmí mít nastaven suid nebo sgid bit
- cgi script musí existovat a být čitelný

Napište třídy stavových kódů protokolu HTTP.

- 1xx infromation
- 2xx success
- 3xx redirection
- 4xx client error
- 5xx server error

Jak probíhá algoritmus výběru hosta při spuštěném ip-based i name-based virtualhostingu?


VIRTUAL HOSTING
NameVirtualHost == IP
+
−
NAME-BASED
ServerName || ServerAlias == Host
+
−
virtual host
1. virtual host
IP-BASED
VirtualHost == IP
+
−
virtual host
default :*
+
−
default host
hlavní server

Y36AWS Teoretická zkouška 12.1.2009

Z STM Wiki

Co je URI a z čeho se skládá, uveďte schéma při http.

URI (celým názvem Uniform Resource Identifier – „jednotný identifikátor zdroje“) je řetězec znaků s definovanou strukturou, který slouží k přesné specifikaci zdroje informací (ve smyslu dokument nebo služba), hlavně za účelem jejich použití pomocí počítačové sítě, zejména Internetu.

protokol://subdomena.subdomena.domena.toplevel:port/cesta/k/souboru/nazev*

protokol://subdomena.subdomena.domena.toplevel:port/cesta/k/souboru/nazev?parametr1=hodnota1&param

*POKRAČ. ? parametr 1 = hodnota 1 & parametr 2 = hodnota 2

Uveďte vztah mezi webspace a filesystémem.

Filesystem je umisteni souboru v souborovem systemu operacniho systemu serveru.

Webspace je system souboru, jak ho vidi klient - adresuje se pomoci URL.

Webspace se konfiguraci serveru mapuje na filesystem.

Co je name-based virtual hosting, jak funguje a proč nefunguje v HTTP/1.0

Name based webhosting umozňuje běh více virtuálních hostů na jedné IP adrese. V HTTP 1.1 prohlížeč posílá hlavičku Host, ve které je zadána jakou URL uživatel zadal. Podle DNS se požadavak pošle na server, který podle hlavičky Host rozhodne, který virtual host použije. V HTTP 1.0 není hlavička Host, proto nefunguje.

Co je a jak funguje MPM Prefork modul

Tento modul používá pro obsluhu požadavků **procesy**. Na začátku vytvoří hlavní proces StartServers obslužných procesů, které čekají na požadavky. V průběhu činnosti serveru jejich počet server upravuje tak, aby počet čekajících procesů ležel mezi hodnotami *MinSpareServers* a *MaxSpareServers*. Maximální počet současně běžících procesů udává hodnota MaxClients. Procesy běží maximálně tak dlouho, dokud neobslouží *MaxRequestsPerChild* požadavků.

Co je to handler (např. u SetHandler) a jak je typicky implementován.

Handler je funkce nějakého modulu, která zpracovává některý soubor. Nastavuje se buď pro všechny soubory v nějakém adresáři (SetHandler nazev) nebo pro všechny soubory s danou příponou daného serveru/vhosta (AddHandler nazev pripona). Je tedy typicky implementován jako funkce v modulu serveru.

Uveďte minimální výstup CGI aplikace.

Na první řádce specifikován Content-type a druhá řádka prázdná

Výhody a nevýhody implementace vlastního modulu

Moduly pro https se píší v jazyce C. Existují ale moduly, které zpřístupňují metody apache přes API dalším jazykům. Moduly tedy lze psát v téměř libovolném jazyce.

Hlavní nevýhoda přímého přístupu k apachím proměnným je v tom, že snadno můžeme podělat celý server :) Problematické je také ladění.

Psát vlastní moduly se tedy vyplatí pouze v případě, že vyžadujeme co nejrychlejší zpracování na straně serveru.

Výhody a nevýhody u php jako modul, CGI a CLI.

PHP jako modul je nejpoužívanější možnost, slouží k dynamickému generování webového obsahu.

PHP jako CGI umožňuje psát CGI-BIN skripty v syntaxi PHP. Slouží rovněž ke generování webového obsahu.

PHP jako CLI umožňuje psát shellové skripty pomocí PHP. Výstup ale není primárně pro web, tzn. nemusí být na prvním řádku Content-type atd. Skript může pracovat se standartním vstupem, výstupem a err výstupem. Na rozdíl od CGI může zpracovávat parametry z příkazové řádky. Nicméně pro psaní shellových skriptů se PHP kvůli rychlosti moc nehodí.

Podrobně popište forward proxy

- Dopředný proxy server je umístěn mezi uživateli a vzdálenými servery.
- Více požadavků na stejný zdroj může být obslouženo z cache (vyšší rychlost).
- Uživatelé musí mít pro používání proxy serveru správně nastavené aplikace.
- Všechen provoz jde zkrz proxy server, proto je možné i řízení přístupu a minitorování dat.

Když je nastaveno Order allow, deny uveďte, jestli bude přístup odepřen nebo povolen, když je shoda

- jen allow - **Povolen**
- jen deny- **Zakázán**
- oba - **Zakázán**
- žádný- **Zakázán**

Rozdíly (nejméně 3) http/1.0 a http/1.1

- "Host" hlavicka pro name-based vhosty
- keepalive
- chunked encoding.

Vysvětlit princip fungování suexec

SuEXEC umoznuje spoustet CGI aplikace pod uzivatelem a skupinou ktera je vlastni, namisto uzivatele/skupiny pod kterou bezi webserveru. Funguje tak, ze webserver namisto CGI aplikace pousti SuEXEC wrapper, ktery je nastaven SUID root (tedy bude bezet jako root, nezavisle na tom kdo ho spustil), a preda mu v ramci parametru cestu k CGI aplikaci. SuEXEC wrapper udela hafomoc bezpecnostnich kontrol a pak, projdou-li, spusti tu aplikaci pod uzivatelem a skupinou ktera jej vlastni.

Vysvětlit princip fungování reverzni proxy

Reverzni proxy funguje na strane poskytovatele obsahu (= webserveru). Funguje tak, ze ji dorazi HTTP pozadavek, ona si ho precte, zpracuje a preposle na nektery z webserveru co sedi za ni. O nicem z tohohle nema klient poneti. Reverzni proxy umoznuje loadbalancing, cachovani, SSLko, komprimaci dat atd. Vyhodou je moznost hodne jemny konfigurace (podle hlavicek pozadavku atd.) oproti loadbalancingu na nizsich vrstvach OSI a spousta peknejch featur, nevyhodou je pak o dost vetsi narocnost (vzit celej pozadavek, precist ho, zpracovat, upravit a poslat na dalsi stroj je narocna cinnost).

K čemu sou filtry a jaký sou jejich typy

Vstupni a vystupni. Umoznuji upravovat data, jako napr. pridavat neco pred ne ci za ne, menit velikost pismen, komprimovat jej, zpracovavat SSI apod.

Pro požadavek http://www.example.org/old se aplikuje rewrite "RewriteRule /old(.*) /new$1 [R]", co se stane a jakymi dalsimi metodami toto jde zaridit?

Provede HTTP redirect na adresu http://www.example.org/new. Lze pres "RedirectMatch /old (.*) /new$1". Take pres nejakou server-side aplikaci.

Vysvětlit HTTP Basic Authentication

Pozadavek neautorizovaneho navstevnika je odmitnut s "401 Authorization required". Klient posle v hlavicce v plaintextu (kodovanym base64) uzivatelske jmeno a heslo. Pokud odpovida, zobrazi se mu pozadovana stranka. Pokud ne, je odmitnut s "403 Forbidden" (nebo teda muze dostat dalsi vyzvu).

Popsat algoritmus pro IP-based virtual hosting

Pozadavky se rozdeluji podle cilove IP adresy pozadavku. Pokud pro danou adresu neexistuje nakonfigurovany virtualhost, pouzije se virtualhost "_default_". Neexistuje-li ani ten, pouzije se globalni server config.

Popsat obecně syntax SSI

```
<!--#element argument="value" argument2="othervalue" ... -->
```

Argumenty jsou oddeleny whitespacem a mohou byt v uvozovkach.

Y36AWS Teoretická zkouška 27.1.2009

Z STM Wiki

Písemka se píše hned po praktické části, jak máte hotovo opravuje vám ji většinou pan Kadlec s vámi a ptá se na případné nejasnosti.

Popište obecnou strukturu HTTP požadavku.

```
Method RequestUri HTTPVersion CRLF
[Header CRLF]*
[CRLF body]
```

Příklad:

```
GET /index.html HTTP/1.1
Host: www.example.com
```

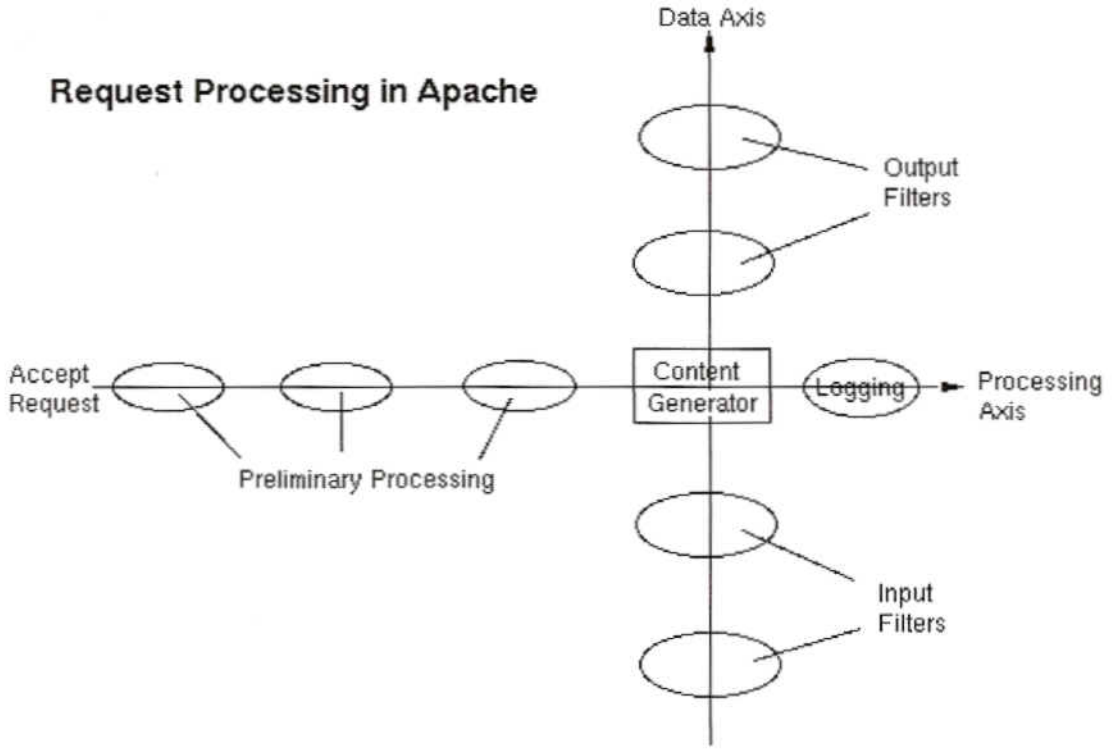
K čemu slouží konfigurační sekce(kontejnery), jaké existují a v čem se liší?

Kontejnery umožňují podmíněnou konfiguraci a omezení platnosti direktiv.

Existují dva druhy podle způsobu vyhodnocování:

- při startu serveru - <IfDefine>, <IfModule> a <IfVersion>
- při požadavku na zdroj - <Directory>, <DirectoryMatch>, <Files>, <FilesMatch>, <Location>, <LocationMatch>, <VirtualHost>, <Proxy>, ...

Popište jak webserver zpracovává požadavek



Popište jak server vyhodnocuje přepisu URL pomoci mod_rewrite

Viz minulá otázka plus:

mod_rewrite provádí dva druhy náhrad. První hned po přijetí požadavku - to pokud je RewriteRule v konfiguraci přímo serveru nebo virtual hosta. Pokud je ale RewriteRule v .htaccess souboru, náhrada se posílá až v pozdější fázi - když je původní URL přetransformována na cestu ve filesystemu. Pokud je v souboru RewriteRule, provede se substituce a je znovu zavolána nová URL (interně). To zpomaluje běh serveru.

Co je to AP, APR?

AP je Apache High-level function, funkce vysokoúrovňového API Apache. Začínají "ap_"

APR je Apache Portable Runtime, API nezávislé na operačním systému. Funkce začínají na "apr_"

Uveďte algoritmus výběru hostitele v případě, že je použit zároveň IP a name based virtual hosting.

Jak zpracovává požadavek FastCGI

Ve FastCGI životní cyklus procesu nekončí s ukončením obsluhy požadavku, ale proces je recyklován a znovupoužit pro obsluhu dalších požadavků. FastCGI aplikace mohou být jedno- i více-vláknové!

1. Webserver (FastCGI process manager) vytvoří procesy aplikace, aby mohl obsluhovat požadavky.
   - při startu serveru - statické aplikace,
   - při příchodu požadavku - dynamické aplikace.
2. FastCGI aplikace provede inicializaci a čeká na požadavky - spojení - od webserveru.
3. Při příchodu požadavku na webserver je vytvořeno spojení s FastCGI procesem a jsou mu odeslány potřebné informace (ty z proměnných prostředí).
4. FastCGI proces přes stejné spojení odešle odpověď (standardní výstup a standardní chybový výstup).
5. Uzavřením spojení mezi webserverem a FastCGI aplikací je obsluha kompletní. FastCGI proces ale nekončí, čeká na další požadavky.

Co je to MPM a k čemu slouží?

Multi Processing Modules - moduly pro souběžné zpracování požadavků slouží pro kontrolu vytváření potomků (procesů a vlákan) k obsluze požadavků na určených portech.

Existují druhy:

- Unix
  - prefork - obsluha pomocí procesů
  - worker - obsluha pomocí vláken (v procesech)
- další platform-specific mpm

Bude přístup odepřen či povolen v případě, že je Order

Deny,Allow a nastane shoda:

- jen allow - **Povolen**
- jen deny- **Zakázán**
- oba - **Povolen**
- žádný- **Povolen**

K čemu jsou a jak fungují *type maps*.

Type Map je soubor (obvykle s příponou var), ve kterém je uloženo jak postupovat při vyjednávání obsahu. Pokud je např. zavolana stranka index.html, ktera neexistuje a existuje soubor index.var, bude se postupovat podle nej. Zde je ulozeno pri jakem jazyku, content-type atd. se zavola jaka jina stranka.

Varianta ?

Klasika vsechno za 4 body. Pisu to trochu jako brainstorming, tak snad se na to po me nekdo vrhne.

1. Popište rozdíly HTTP/1.0 a 1.1.

hlavička host, keep-alive, chunked (chce to malinko rozvést).

2. Popište vlastnosti souborů .htaccess. Jaké jsou jejich výhody a nevýhody.

Upravují volby povolené přes AllowOverride v konfiguraci. Usnadňují konfiguraci přímo v adresářích. Načítají se při každém dotazu.

3. Uveďte algoritmus výběru hostitele v případě, že je použit zároveň IP a name based virtual hosting.

Notoricky známý obrázek.

4. nemůžu si vzpomenout

5. multiviews

6. co znamena Q?? R L u Re...

nepamatuju přesně otázku.. tohle jsem neměl.

7. co dela Re... http://www.example.com/old(.*) /new$1 [R] a jak jinak by se to dalo zaridit.

viz. starší ročníky

8. popsat a oduvodnit 4 kontroly suexec

Script ani nadřazený adresář není writable pro nikoho jiného než vlastníka příp. skupinu - aby nemohl script upravit ani přepsat jiný uživatel. Script není SUID nebo SGID - aby nedělal neplechu

Script existuje - aby bylo co zpuštět ...

9. prava pristupu.. byla tabulka kdo se kam snazi prihlasit a konfigurace nejak takhle:

```
Location adresar
 basic autorizace
 restriction valid-user
 order deny,allow
 allow *.mycorp.k328
 match any
Location podadresar
 match all
 restrict user admin nekdo
```

v tabulce byly snad vsechny moznosti z adresar/podadresar, prihlasen/neprihlasen, z povolene domeny/z nepovolene domeny

10. Jaký je nejjednodušší výstup CGI scriptu?

Content-type: <mime-typ><CRLF> <CRLF>