Command injection in iRules

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

Who am I and thanks

Big thanks to my fellow researchers

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

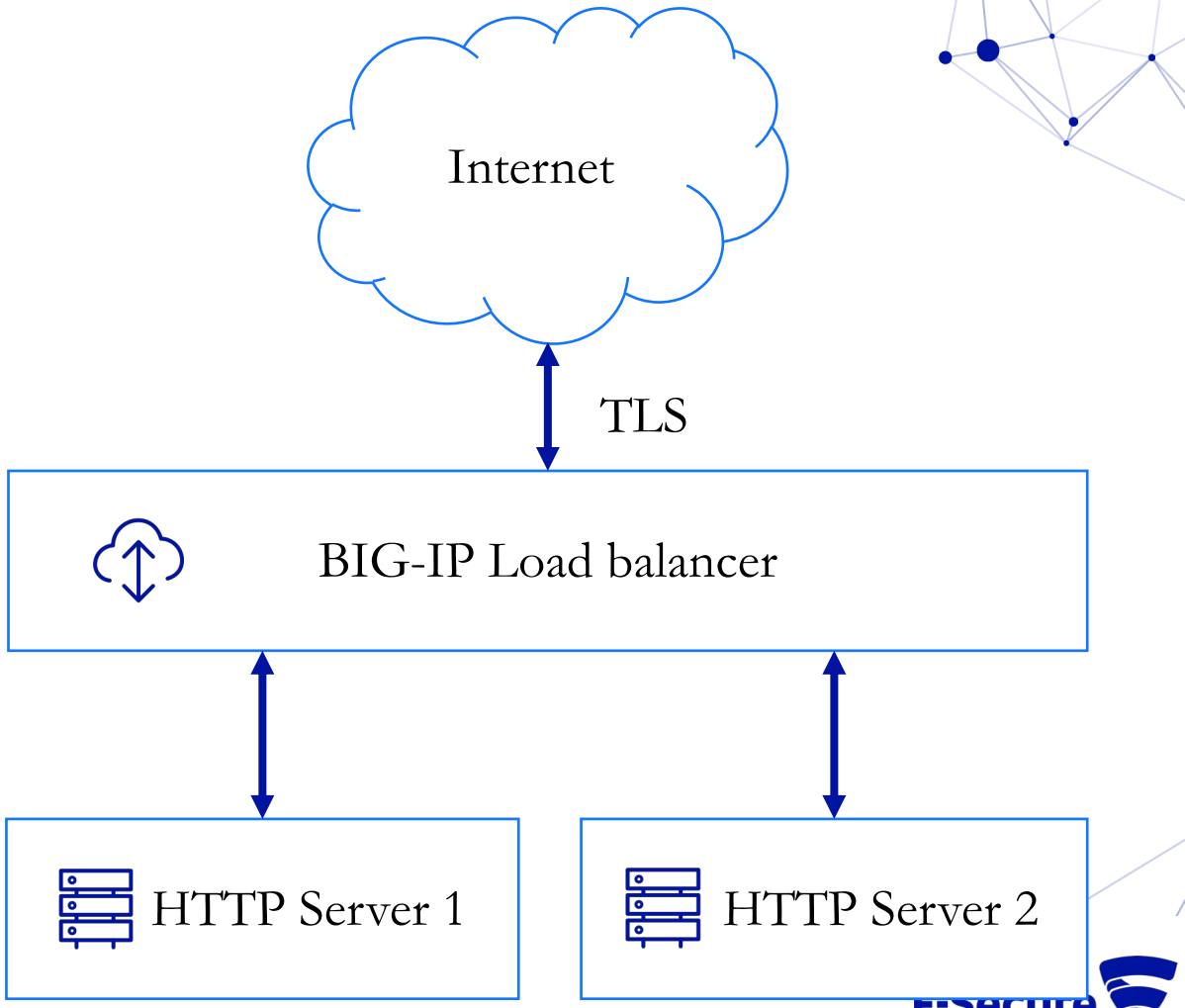
Big thanks to David and Aaron at F5 SIRT for a good response
https://support.f5.com/csp/article/K15650046

F-Secure

What F5 products we are talking about

- Can store and handle multiple sessions for backend servers
- A customer will write their own iRules to define the laod balancer behaviour
- Devcentral is used as a stackoverflow of 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



What iRules are

- A fork of TCL 8.4
  - New features in TCL 8.4< are not introuduced in iRule
  - iRule have introduced a group of simplifications and exceptions to TCL
  - Return oriented programming (with the option of rarely used exception handling)

TCL / iRule basics

- iRules determine where a given HTTP request is forwarded based on input criteria
  - The HTTP request header and body is parsed by the iRule F5 engine
  - The system admnistrator write F5 iRule code to handle a request
  - devcentral.f5.com contain example scripts commonly used
- Example:

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

How to spot them in the wild

- HTTP header include
  - Server: BigIP
    - Found in redirects
    - Found in favicon.ico response

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

F-Secure

Map of usage based on server header search

- Countries ▼ +

United States (us)	180,246
Germany (de)	14,083
Australia (au)	11,969
United Kingdom (gb)	11,068
Canada (ca)	10,516
France (fr)	8,406
Japan (jp)	6,942
Sweden (se)	5,261
China (cn)	5,258
Netherlands (nl)	3,303

F-Secure

This is 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.)
Moe: Wait a minute... Listen, you little yellow-bellied rat jackass, if I ever find out who you are, I'm gonna kill you!

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
```

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 bash

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

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

List of builtin commands that can perform command evaluation

- after
- catch
- eval
- for
- foreach
- history
- if
- proc
- cpu
- string match
- interp
- namespace eval
- namespace inscope
- source
- switch
- 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

Starting with ”-crash”

switch ?*options*? *string pattern body* ?*pattern body* ...?

```
switch  $x
 "ONE" "puts ONE=1”
 "TWO" "puts TWO=2"
 "default" "puts NO_MATCH";
```

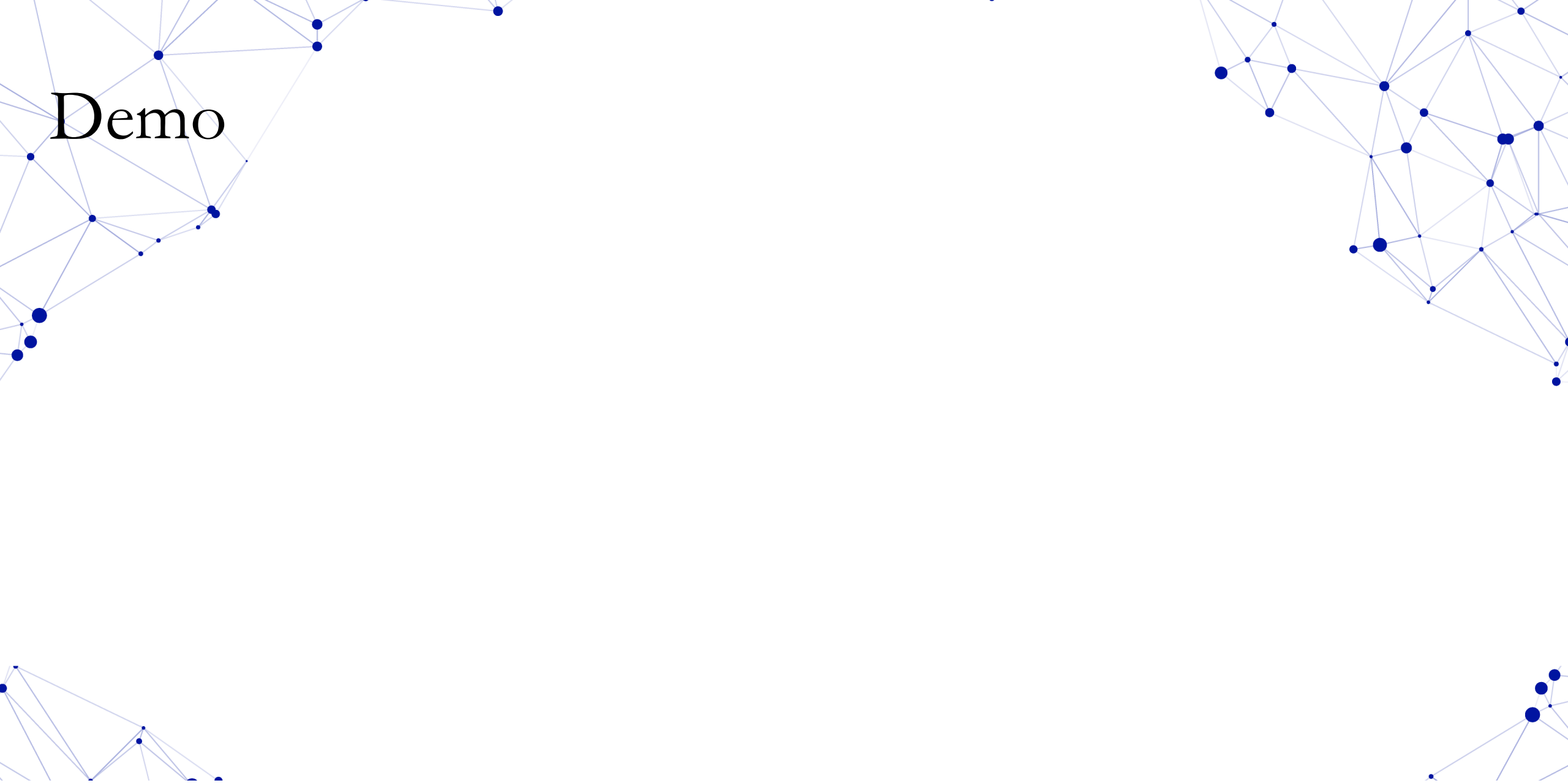
- When the user controls $x the input can be both an option or the match string
- If the user input `-crash` or any illegal option the switch statement will crash and the iRule will not continue
- A `-regex` or `-glob` option can also be used to manipulate the outcome of the switch statement
- This circumvent logging and is detected when the page times out due to the crash
- The attack is great for avoiding detection because the request is not forwarded or logged

F-Secure

Exploitation

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

Demo



Gaining permanent access using session table

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

Hacking the session table

- With command injection its 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

Demo session table

A look at the code from 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

Scanning for command injection

- Automated tool to find quoted and unquoted arguments
- Its unmaintained rust so I had to fix it
- Get the code of tclscan https://github.com/kugg/tclscan

Unit testing the iRule code using testcl

- Get it at 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 more vulnerabilities
- Demo of testcl on devcentral snippets