

## Computing HMAC

### Use Case:

HMAC is a keyed-hash message authentication code computed involving a cryptographic hash function in combination with a secret cryptographic key. It can be used to simultaneously verify both the data integrity and the authentication of a message. When we deploy the ADC solution is secure and data sensitive use cases, HMAC computation on data flowing through is needed by customers.

### F5 iRules:

```
when RULE_INIT {  
  
    set message "test"  
    set input { "1234" \  
                "123456789012345678901234567890123456789012345678901234  
567890xxxx" \  
                "yyyy123456789012345678901234567890123456789012345678901234567890  
1234567890xxxx" \  
    }  
  
    foreach prekey $input {  
        set bsize 64  
        if { [string length $prekey] > $bsize } {  
            set key [sha256 $prekey]  
        } else {  
            set key $prekey  
        }  
  
        set ipad ""  
        set opad ""  
        for { set j 0 }{ $j < [string length $key] }{ incr j }{  
            binary scan $key @$j}H2 k  
            set o [expr 0x$k ^ 0x5c]  
            set i [expr 0x$k ^ 0x36]  
            append ipad [format %c $i]
```

```

        append opad [format %c $o]
    }

    for { }{ $j < $bsize }{ incr j }{
        append ipad 6
        append opad \\
    }

    set token [sha256 $opad[sha256 "${ipad}${message}"]]

    binary scan $token H* hextoken
    log -noname local0. [string toupper "result = $hextoken"]
}

}

```

### [NetScaler Solution:](#)

```

add hmac h1 -digest SHA1 -keyValue 010203040500000000

add rewrite action act1 replace HTTP.REQ.HEADER("TEST")
HTTP.REQ.HEADER("TEST").HMAC("h1")

add rewrite policy pol1 HTTP.REQ.HEADER("TEST").EXISTS act1

```

NetScaler advance policies support HMAC operation as an independent function which can just be used with any data set. This simplifies the whole use case and one need not worry about complex calculations.