Práctica: Ping en binario, decimal, octal e hexadecimal

Paso 5. *Que acontece? Por que? Razoa a resposta.*
O comando ping comproba sempre a conectividade coa IPv4 127.0.0.1, non sendo o primeiro comando (ping -c4 01111111.00000000.00000000.00000001), xa que non é capaz de entender o código binario.

Paso 7. *Que acontece? Por que? Razoa a resposta.*
Acontece o mesmo que en sistemas GNU/Linux.

Paso 8. *Facer o mesmo procedemento anterior coa IPv4 10.0.2.15 (NAT de VirtualBox)*

a. Binario:
10 -> $(00001010)_2$
0 -> $(00000000)_2$
2 -> $(00000010)_2$
15 -> $(00001111)_2$

Polo tanto 10.0.2.15 equivale a:
$(00001010.00000000.00000010.00001111)_2$ -> $(00001010000000000000001000001111)_2$

b. Decimal:
Convertir o anterior número binario a decimal:
$(00001010000000000000001000001111)_2$ -> 167772687

Polo tanto 10.0.2.15 equivale a 167772687

c. Octal:
Convertir o anterior número binario en octal:
$(00001010000000000000001000001111)_2$ -> $(1200001017)_8$ -> 1200001017

Pero tamén poderíamos facer as seguintes conversións:
10 -> $(12)_8$ -> 012
0 -> $(0)_8$ -> 00
2 -> $(2)_8$ -> 02
15 -> $(17)_8$ -> 017

Polo tanto 10.0.2.15 equivale a 0120002017 ou 012.00.02.017

d. Hexadecimal:
Convertir o anterior número binario en hexadecimal
$(00001010000000000000001000001111)_2$ -> $(A00020F)_{16}$ -> 0xA00020F -> 0xa00020f

Pero tamén poderíamos facer as seguintes conversións:
10 -> $(A)_{16}$ -> 0xA -> 0xa
0 -> $(0)_{16}$ -> 0x0

Ricardo Feijoo Costa

2 -> $(2)_{16}$ -> 0x2
15 -> $(F)_{16}$ -> 0xF -> 0xf

Polo tanto 10.0.2.15 equivale a 0xA02F ou 0xA.0x0.0x2.0xF ou 0xa.0x0.0x2.0xf

Probamos distintos comandos ping:
Binario -> ping -c4 00001010000000000000001000001111 -> Non resolve, xa que ping non entende o código binario e interpreta ese número como decimal.
Decimal -> ping -c4 167772687 -> Resolve (Interpreta o número como decimal)
Decimal -> ping -c4 0167772687 -> Non resolve (Interpreta o número como octal)
Octal -> ping -c4 1200001017 -> Non resolve a IPv4 10.0.2.15 (Interpreta o número como decimal)
Octal -> ping -c4 01200001017 -> Resolve (Interpreta o número como octal)
Octal -> ping -c4 012.00.02.017 -> Resolve (Interpreta os números entre puntos como octal)
Hexadecimal -> ping -c4 A00020F -> Non resolve (Interpreta o número coma nome de equipo)
Hexadecimal -> ping -c4 0xa.0x0.0x2.0xf -> Resolve (Interpreta os números entre puntos como hexadecimal)

Paso 9. *Que acontece? Por que? Razoa a resposta.*

Entón, concluímos que o comando ping pode intentar resolver calquera número, tal que:

1) Se o número non comeza por un cero interpreta ese número coma decimal. Exemplos:

167772687 que equivale a 10.0.2.15 en base decimal

127.0.0.1 que xa está en formato IPv4 en base decimal

2) Se o número comeza por un cero interpreta ese número coma octal. Exemplos:

01200001017 que equivale a 10.0.2.15 en base decimal

0177.00.00.01 que equivale a 127.0.0.1 en base decimal

3) Se o número comeza por un 0x interpreta ese número como hexadecimal. Exemplos:

0xa.0x0.0x2.0xf que equivale a 10.0.2.15 en base decimal

0x7f.0x0.0x0.0x1 que equivale a 127.0.0.1 en base decimal

Paso 10. *Facer o mesmo procedemento anterior coa IPv4 172.217.168.163 (posible dirección de www.google.es)*

a. Binario:
172 -> $(10101100)_2$
217 -> $(11011001)_2$
168 -> $(10101000)_2$
163 -> $(10100011)_2$

Polo tanto 172.217.168.163 equivale a:

Ricardo Feijoo Costa

$(10101100.11011001.10101000.10100011)_2$ -> $(10101100110110011010100010100011)_2$

b. Decimal:
Convertir o anterior número binario a decimal:
$(10101100110110011010100010100011)_2$ -> 2899945635

Polo tanto 172.217.168.163 equivale a 2899945635

c. Octal:
Convertir o anterior número binario en octal:
$(10101100110110011010100010100011)_2$ -> $(25466324243)_8$ -> 25466324243

Pero tamén poderíamos facer as seguintes conversións:
172 -> $(254)_8$ -> 0254
217 -> $(331)_8$ -> 0331
168 -> $(250)_8$ -> 0250
163 -> $(242)_8$ -> 0243

Polo tanto 172.217.168.163 equivale a 0254033102500243 ou 0254.0331.0250.0243

d. Hexadecimal:
Convertir o anterior número binario en hexadecimal
$(10101100110110011010100010100011)_2$ -> $(ACD9A8A3)_{16}$ -> 0xACD9A8A3 -> 0xacd9a8a3

Pero tamén poderíamos facer as seguintes conversións:
172 -> $(AC)_{16}$ -> 0xAC -> 0xac
217 -> $(D9)_{16}$ -> 0xD9
168 -> $(A8)_{16}$ -> 0xA8
163 -> $(A3)_{16}$ -> 0xA3

Polo tanto 172.217.168.163 equivale a 0xACD9A8A3 ou 0xAC.0xD9.0xA8.0xA3 ou 0xac.0xd9.0xa8.0xa3

Probamos distintos comandos ping:
Binario -> ping -c4 10101100110110011010100010100011 -> Non resolve, xa que ping non entende o código binario e interpreta ese número como decimal.
Decimal -> ping -c4 2899945635 -> Resolve (Interpreta o número como decimal)
Decimal -> ping -c4 02899945635 -> Non resolve (Interpreta o número como octal)
Octal -> ping -c4 25466324243 -> Non resolve (Interpreta o número como decimal)
Octal -> ping -c4 025466324243 -> Resolve (Interpreta o número como octal)
Octal -> ping -c4 0254.0331.0250.0243 -> Resolve (Interpreta os números entre puntos como octal)
Hexadecimal -> ping -c4 ACD9A8A3 -> Non resolve (Interpreta o número coma nome de equipo)
Hexadecimal -> ping -c4 0xAC.0xD9.0xA8.0xA3 -> Resolve (Interpreta os números entre puntos como hexadecimal)

Ricardo Feijoo Costa

Paso 11. *Que acontece? Por que? Razoa a resposta.*

Entón, concluímos que o comando ping pode intentar resolver calquera número, tal que:

1) Se o número non comeza por un cero interpreta ese número coma decimal. Exemplos:

    2899945635 que equivale a 172.217.168.163 en base decimal

    127.0.0.1 que xa está en formato IPv4 en base decimal

2) Se o número comeza por un cero interpreta ese número coma octal. Exemplos:

    0254.0331.0250.0243 que equivale a 172.217.168.163 en base decimal

    0177.00.00.01 que equivale a 127.0.0.1 en base decimal

3) Se o número comeza por un 0x interpreta ese número como hexadecimal. Exemplos:

    0xAC.0xD9.0xA8.0xA3 que equivale a 172.217.168.163 en base decimal

    0x7f.0x0.0x0.0x1 que equivale a 127.0.0.1 en base decimal

Paso 12. *Lanza o navegador Firefox e proba se coas IPs do paso 4 é posible navegar. Exemplo: http://0x7F.0x1*

01111111.00000000.00000000.00000001 -> Non resolve
2130706433 -> Resolve
017700000001 -> Resolve
0177.00.00.01 -> Resolve
0x7F000001 -> Resolve
0x7f000001 -> Resolve
0X7f000001 ->Resolve
0x7f.0x0.0x0.0x1 -> Resolve
0x7F.0x1 -> Resolve

Paso 13. *Que acontece? Por que? Razoa a resposta.*

01111111.00000000.00000000.00000001 -> Non resolve (Interpreta o número como decimal)
2130706433 -> Resolve (Interpreta o número como decimal: 127.0.0.1)
017700000001 -> Resolve (Interpreta o número como octal equivalente ao decimal 127.0.0.1)
0177.00.00.01 -> Resolve (Interpreta o número como octal equivalente ao decimal 127.0.0.1)
0x7F000001 -> Resolve (Interpreta o número como hexadecimal equivalente ao decimal 127.0.0.1)
0x7f000001 -> Resolve (Interpreta o número como hexadecimal equivalente ao decimal 127.0.0.1)
0X7f000001 ->Resolve (Interpreta o número como hexadecimal equivalente ao decimal 127.0.0.1)
0x7f.0x0.0x0.0x1 -> Resolve (Interpreta o número como hexadecimal equivalente ao decimal 127.0.0.1)
0x7F.0x1 -> Resolve (Interpreta o número como hexadecimal equivalente ao decimal 127.0.0.1)

Ricardo Feijoo Costa

Exercicio12: URL Encode

Paso 2. *Que acontece?*

Visítase a páxina https://www.debian.org

Paso 4. *Que acontece?*

Visítase de novo a páxina https://www.debian.org

Paso 5. *Executa o seguinte comando:*

$ echo %77%77%77%2e%64%65%62%69%61%6e%2e%6f%72%67 | xxd -r -p ; echo

Que acontece?

Amósase a cadea de caracteres: www.debian.org, xa que o comando *xxd* é capaz de convertir código hexadecimal en caracteres ASCII, sendo %NM unha forma de representar o código hexadecimal 0xNM nos navegadores (protocolos HTTP, HTTPS...)

Paso 8. *Identifica os caracteres ASCII correspondentes aos seguintes valores hexadecimais:*

HEX	ASCII
0x77 ->	w
0x77 ->	w
0x77 ->	w
0x2e ->	.
0x64 ->	d
0x65 ->	e
0x62 ->	b
0x69 ->	i
0x61 ->	a
0x6e ->	n
0x2e ->	.
0x6f ->	o
0x72 ->	r
0x67 ->	g

Paso 9. *Une os anteriores caracteres. Que acontece?*

www.debian.org , sería a URL que lanzamos anteriormente no navegador.

Paso 10. *Agora na táboa ASCII identifica os valores hexadecimais dos seguintes caracterers ASCII: www.kernel.org*

ASCII	HEX
w ->	77
w ->	77
w ->	77
. ->	2e
k ->	6b
e ->	65
r ->	72
n ->	6e
e ->	65
l ->	6c
. ->	2e

Ricardo Feijoo Costa

```
o    ->  6f
r    ->  72
g    ->  67
```

Paso 11. *A cada valor HEX atopado do punto anterior:*

a. *Precédelle un caracter %, é dicir, se atopas 65 escribe %65.*
   %77 %77 %77 %2e %6b %65 %72 %6e %65 %6c %2e %6f %72 %67

b. *A continuación une todos os valores sen espazo.*
   %77%77%77%2e%6b%65%72%6e%65%6c%2e%6f%72%67

c. *Agora visita a dirección http://valoratopado, onde valor atopado serán os valores hexadecimais cos % do paso anterior. Así, se por exemplo atopas, %65%66 debes visitar a dirección http://%65%66*
   http://%77%77%77%2e%6b%65%72%6e%65%6c%2e%6f%72%67

d. *Que acontece? Por que? Razoa a resposta.*
   Visítase a páxina web www.kernel.org

Ricardo Feijoo Costa