

Giorna Linux 2.0

*Just keep
on Hacking*



**Ruby: una
gemma di
linguaggio /2**

QR-Code

MITM

Wiimote



L'Uomo e la Terra

Andy <andrea.turconi@gmail.com>

DA TEMPO il problema del *Climate Change* e i conseguenti ragionamenti sui futuri modelli di sviluppo della società umana tengono banco sulla scena scientifica e culturale, ma è forse solo con la presente crisi economica che questi temi sono entrati nell'agenda politica internazionale. Da una parte si è fatta strada l'idea che la *green economy* possa costituire un volano per superare la crisi investendo in modo lungimirante per il futuro. Dall'altra l'interrogarsi sulle spirali deleterie (divario Nord-Sud del Mondo, inquinamento, ...) innescate dal modello socio-economico entrato in crisi, ha reso di attualità le riflessioni su giustizia sociale ed ecologia. Tra altre iniziative è tornata in auge la necessità dell'abbassamento delle emissioni di CO₂.

I principali avversari in questa polemica sono gli "ambientalisti-irriducibili", per i quali le dichiarazioni dell'IPCC sono "Vangelo", e i "depositari-della-Vera-Scienza" che con altrettanta mancanza di laicità non vedono l'ora di smontare teorie, studi e modelli per dimostrare l'insignificanza dell'impatto umano sul clima, spesso per assolvere l'attuale modello economico.

A fronte dell'ancora incompleta compren-

sione del fenomeno, conviene ragionare con i principi della prudenza e del male minore poiché la Terra è, per il momento, l'unico pianeta conosciuto che ospiti la vita; inoltre il numero di variabili per la modellazione del clima è molto alto, le loro interconnessioni oltremodo intricate, e per di più non siamo a conoscenza di tutti i meccanismi in gioco.

Basterebbe allora limitarci a constatare che, a prescindere dai cambiamenti naturali che sono sempre presenti, la specie umana ha arrecato nel suo espandersi tanti più danni all'ambiente – e a se stessa – quanto più ha dimenticato i limiti dell'equilibrio dell'ecosistema. Ne consegue che è meno grave rallentare la crescita e scoprire che ci sono più risorse, piuttosto che trovarsi alla frutta dopo aver sfruttato tutto lo sfruttabile.




Per un superamento virtuoso della crisi è quindi necessario riuscire tutti ad acquisire la consapevolezza di essere a bordo della stessa "astronave" e che il futuro di ciascuno è legato a quello di tutti gli altri e a quello dello stesso Pianeta.

In una parola... *Ubuntu*: "io sono ciò che sono in virtù di ciò che tutti siamo".

Indice

Ruby vol. 2: affiliamo la katana	3
Codici a barre senza barre	6
Le verità nascoste	8
Utilizzare il Wiimote su Linux	11

Quest'opera è rilasciata sotto la licenza Creative Commons BY-NC-SA 2.5. Questo significa che sei libero di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire o recitare l'opera e creare opere derivate alle seguenti condizioni:

-  **Attribuzione.** Devi riconoscere il contributo dell'autore originario.
-  **Non commerciale.** Non puoi usare quest'opera per scopi commerciali.
-  **Condividi allo stesso modo.** Se alteri, trasformi o sviluppi quest'opera, puoi distribuire l'opera risultante solo per mezzo di una licenza identica a questa.

In occasione di ogni atto di riutilizzazione o distribuzione, devi chiarire agli altri i termini della licenza di quest'opera. Se ottieni il permesso dal titolare del diritto d'autore, è possibile rinunciare ad ognuna di queste condizioni.

Le tue utilizzazioni libere e gli altri diritti non sono in nessun modo limitati da quanto sopra.

Questo è un riassunto in linguaggio accessibile a tutti del Codice Legale:

<http://creativecommons.org/licenses/by-nc-sa/2.5/it/legalcode>



Ruby vol. 2: affiliamo la katana

Simone D'Amico <sim@me.com>

NELLA SCORSA PUNTATA abbiamo fatto una breve digressione storica su questo superbo linguaggio. Abbiamo inoltre preparato gli strumenti base (*irb*, *ri*, *ruby*) per la nostra prima spedizione speleologica. Armati di tutto punto, con il nostro editor preferito che blinkava in palpitazione, ci siamo però lasciati senza vedere una sola riga di codice.

Oggi invece vedremo come vengono utilizzati in ruby gli elementi principali della programmazione, darò per scontati una conoscenza pregressa di un linguaggio di programmazione e un know-how informatico di base.

Irb

irb è l'interprete Ruby interattivo, che specialmente nella fase di apprendimento del linguaggio o nella fase iniziale di stesura di un programma ci sarà parecchio utile.

La principale differenza dall'interprete Ruby normale è che permette di stendere codice "a run-time" e di stampare alla fine di ogni singola istruzione un'inspect dell'oggetto su cui stavamo lavorando, levandoci il fastidio di scrivere un'ulteriore istruzione

di stampa per avere conferma che tutto sia andato per il verso giusto.

A conferma della solidità e dell'orientamento alla sicurezza dell'interprete comandi vi è la sua versione online di cui vi ho accennato nello scorso articolo e che ora è disponibile in veste rinnovata all'indirizzo <http://TryRuby.org>



I tipi base

Una moderna guida a un linguaggio di programmazione privilegerebbe un approccio passo-passo e graduale. Tuttavia visto l'esiguo spazio rimando alla bibliografia Ruby (in particolare il Pickaxe), nel caso durante il corso non riusciste a dileguare i vostri dubbi con l'ampia documentazione online. Passando subito al sodo vediamo quali sono i tipi base forniti dal linguaggio e che

probabilmente ci ritroveremo a manipolare più spesso.

Interi e numeri a virgola mobile

I numeri in Ruby sono istanze principalmente di due classi: `FIXNUM` e `BIGNUM`. Un `FIXNUM` contiene numeri che è possibile contenere in una parola di memoria (meno un bit); quando un numero eccede questo spazio è automaticamente convertito in un `BIGNUM` che è limitato unicamente dalla memoria disponibile. Le costanti letterali possono essere scritte indicando il segno e la base, oltre ad avere la possibilità di utilizzare la notazione scientifica. Numeri razionali e complessi hanno classi apposite nella libreria standard, ma non vi è un supporto a livello di linguaggio come nei numeri interi.

Stringhe

Accontentandoci di una descrizione superficiale (avremo modo di approfondirle in articoli seguenti), le stringhe sono porzioni di testo delimitato da apici singoli o doppi (la differenza sta nelle sostituzioni che si vanno ad operare all'interno della stringa), oppure utilizzando un carattere a nostra scelta con `%q` o `%Q`: ad esempio utilizzando le parentesi graffe potremmo scrivere, senza il timore di incorrere in sostituzioni, così:

```
%Q{Petrarca disse: "Erano i capei d'oro a l'aura sparsi"}
```

Intervalli

Sono istanze della classe `RANGE` e sono spesso usate nelle espressioni condizionali. Due punti sono usati per intervalli chiusi a destra, mentre tre punti per intervalli aperti a destra.

Un modo molto semplice per ottenere una array con le lettere da `m` a `q`, con `q` esclusa è il seguente:

```
lettere = ('m'...'q').to_a
=> ["m", "n", "o", "p"]
```

Array

Per costruire un array è sufficiente racchiudere in parentesi quadre un insieme di oggetti separati da virgole. Esistono anche qui zuccheri sintattici ("syntactic sugar" è il termine che la comunità Ruby usa per definire queste abbreviazioni) per ovviare ai compiti più frequenti.

Ad esempio in modo simile a `%q` è possibile definire un array di stringhe nel seguente modo:

```
bbt = %w{Leonard Sheldon Penny
        Howard Rajesh}
=> ["Leonard", "Sheldon", "Penny",
    "Howard", "Rajesh"]
```

Hash

Sono insiemi di elementi `chiave=>valore` talvolta conosciuti come dizionari. È possibile definirli dentro parentesi quadre separati da virgole come nel seguente esempio:

```
food = {"sausages" => "meat",
        "apples" => "fruit",
        "banana" => "fruit"}
```

Le chiavi devono essere uniche, mentre i valori possono essere duplicati.

E dunque...

Anche per questa puntata è tutto, tenetevi stretta la vostra gemma in attesa del prossimo articolo!

Simboli

Talvolta chiamati atomi in altri linguaggi, sono un modo per utilizzare una stringa non tanto per il valore che essa contiene ma per quanto il suo nome rappresenta. Non è difficile intuire che uno degli utilizzi principali è negli hash.

Un simbolo si definisce con i due punti prima del nome, ad esempio:

```
me = {:nome => "Simone",
      :cognome => %q{D'amico},
      :età => 19}
```

oppure usando un syntactic sugar di Ruby 1.9:

```
me = {nome: "Simone",
      cognome: %q{D'Amico},
      età: 19}
```

Espressioni regolari

Sono istanze di `REGEXP` ed è possibile costruirle raccogliendole dentro degli “smash”. Funzionano come in Perl, ed essendo piuttosto complesse meriteranno un articolo a parte.

Se proprio vi prudono le mani, potete provare questo frammento dal significato autoesplicativo:

```
p "BORP!" if /birra/ =~
'buona la birra?'
```

Codici a barre senza barre

Andreabont <andreabont@gmail.com>

O RMAI SIAMO tutti abituati alla presenza dei codici a barre, sui prodotti, sui libri, sulle tessere e così via, ma non tutti conoscono l'evoluzione di questi codici, che diventano bidimensionali e sostituiscono le barre con dei... puntini!

La codifica

Come per i codici a barre classici, anche quelli bidimensionali presentano diverse codifiche possibili. La codifica che vedremo in questo articolo è il QR-Code (*Quick Response Code*) sviluppato nel 1994 dalla corporation giapponese Denso-Wave.

La codifica è diventata nel 2000 uno Standard Internazionale (ISO/IEC 18004) di fatto aperto, dato che la sua specifica è pubblica e il brevetto posseduto dalla Denso-Wave volutamente non esercitato. Questa codifica è maggiormente usata proprio in Giappone, mentre in Italia è più diffuso di fatto il Data Matrix (in grado di memorizzare più informazioni), codifica molto simile usata per esempio dalle Poste Italiane. Nonostante sia poco diffuso nel nostro paese, mantiene un notevole interesse a causa della sua versatilità: può essere letto con facilità da un qualsiasi dispositivo

dotato di telecamera (computer fissi o portatili, palmari, cellulari ecc...) contenente un apposito software di decodifica.

Software usato

Il software da usare non è facilissimo da trovare, più che altro a causa della miriade di software a pagamento (molto costosi) che rendono la ricerca fastidiosa. Per questo mi permetto di consigliare un software Open Source in grado di leggere i codici a barre (normali o bidimensionali) con differenti codifiche sia da webcam che da immagine salvata.

Il programma si chiama *zbar* (<http://zbar.sourceforge.net>) e nella sua versione 0.10 gira sia su Linux che su sistemi Windows (fortemente sconsigliato ;-)); il pacchetto contiene due programmi distinti

accessibili da linea di comando, *zbarcam* per avviare la lettura da webcam e *zbarimg* per analizzare le immagini salvate.

Un altro programma utile è *zint* (<http://www.zint.org.uk/zintSite>), usato invece per generare i codici.



Dettagli tecnici

Il QR-Code si presenta come una matrice quadrata di punti neri su sfondo bianco (o viceversa) nella quale si può memorizzare una determinata quantità di informazioni a seconda della loro natura e delle dimensioni della matrice stessa.



Proprio per renderlo leggibile dalla maggior parte dei dispositivi esso presenta due principali caratteristiche: la prima è la più evidente, ovvero un sistema visivo di allineamento agli angoli (i 2 quadrati l'uno dentro l'altro) ad esclusione dell'angolo in basso a destra, mentre la seconda non è visibile ed è un sistema di correzione degli errori di lettura che sfrutta l'algoritmo Reed-Solomon, la cui percentuale di successo dipende dalla codifica di partenza (decisa alla creazione) che permette una lettura senza errori anche in mancanza di parti del codice stesso o a causa di difetti dati dalla lettura con dispositivi di bassa qualità.

In generale i QR-Code presentano le seguenti capacità massime di memorizzazione:

- 7089 caratteri (solo numerico);
- 4296 caratteri (alfanumerico);

- 2953 caratteri (binario a 8 bit);
- 1817 caratteri (Kanji/Kana).

La percentuale di ripristino delle parti danneggiate è del:

- 7% (livello L);
- 15% (livello M);
- 25% (livello Q);
- 30% (livello H).

Esiste anche una tecnica chiamata "Grafica QR" che permette di disporre i punti del codice in modo da creare dei semplici disegni, i quali non impediscono la lettura del codice stesso; questo è utile poiché permette all'utente di capire per esempio chi ha creato il codice che andrà a leggere. E' intuibile che l'avanzamento tecnologico ai QR-Code porterebbe dei vantaggi ad una lunga serie di settori tra cui i meccanismi di pagamento, di stoccaggio e di logistica, con conseguente aumento dell'efficienza dei sistemi gestionali.

Link utili

Potete divertirvi a giocare con i codici a barre su questi siti:

- generatore di codici: <http://www.barcode-generator.org>
- lettore di codici: <http://www.onlinebarcodereader.com>

Le verità nascoste

Mario Polino <mpolino90@gmail.com>

VI TERRORIZZA che qualche spione riesca a sapere cosa fate con il vostro PC? Che riesca a sapere esattamente quali siti visitate? Che qualcuno riesca con facilità a rubarvi password e chiavi di accesso? Ci sono innumerevoli pericoli per la vostra privacy e cosa infinitamente più grave per le vostre password. Ottenere i vostri dati può rivelarsi questione di qualche click ben assestato.

Cos'è il MITM?

Il nome MITM (*Man in the Middle*, ovvero *Uomo nel Mezzo*) rende bene l'idea: si tratta infatti di un attacco che mira ad intercettare dati scambiati tra due o più parti.



In questa tipologia l'attaccante riesce a far credere alle vittime di essere il destinatario di ogni trasmissione ricevendo, quindi, tutti i pacchetti. L'attacco non risulta

visibile poiché l'attaccante dopo aver letto e/o modificato i pacchetti provvede ad inoltrarli al reale ed ignaro destinatario.

Potenzialità del MITM

Ora proviamo ad immaginare il seguente scenario apocalittico. Abbiamo adocchiato un oggetto fighissimo su un certo sito di compravendite: è troppo bello! Deve essere nostro! Il pagamento si effettua tramite carte elettroniche, ma la nostra prepagata è scarica (lo è sempre...); abbiamo dunque la brillante idea di utilizzare la carta di credito di papà. Inseriamo nome, cognome, scadenza, codice della carta, codice di sicurezza, codice di codice e ancora un altro codice. Diamo un'altra controllata ai dati e puntiamo il cursore del mouse sul tasto 'Conferma'. Peccato che in questo momento Cereal Killer ¹ è a corto di soldi e quindi decide di fare un bel MITM su tutta la zona, e noi purtroppo ricadiamo nella rete da lui controllata...

Cliccando sul bottone di assenso il nostro browser genera il pacchetto che viene affidato alla scheda di rete e poi trasmesso. La catastrofe è compiuta: il pacchetto arriva a Cereal Killer che ne salva il contenuto

¹Personaggio tratto dal Film "Hackers" (1995)

e lo reinoltra per fargli continuare il suo viaggio fino al server che gestisce tutta la situazione.

A questo punto la transizione è stata effettuata, il nostro amico ha tutti i codici della carta e può quindi anche lui fare acquisti online con la carta del babbo.

Un po' di teoria

Fantascienza a parte, esistono diverse tecniche che permettono di trarre in inganno gateway e PC, ed esistono anche alcuni programmi che rendono il tutto estremamente facile. Una delle tecniche più diffuse in ambienti LAN è sicuramente l'ARP Poisoning.

Questa tecnica sfrutta il poco sicuro protocollo ARP che serve ad assegnare la corrispondenza IP-MAC. Una macchina A per comunicare con un preciso IP associato ad una macchina B ha bisogno di sapere qual è l'indirizzo MAC di B; per questo motivo A invia un pacchetto ARP in cui chiede chi sia B e normalmente questo gli risponde con il suo indirizzo che viene salvato sulla macchina A nella ARP cache.

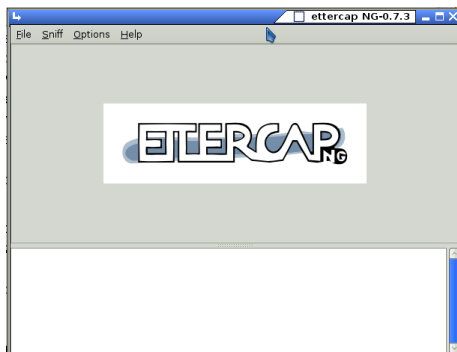
Ora che conosciamo l'idea base del funzionamento del protocollo ARP possiamo farci una domanda: cosa succede se inviamo un pacchetto di risposta ad A senza che questo ne abbia fatto richiesta? In questo caso A aggiorna la sua ARP cache con le informazioni del nuovo pacchetto; basta quindi inviare alla macchina vittima un pacchetto ARP in cui diciamo per esempio che all'indirizzo 192.168.1.1 corrisponde il nostro indirizzo MAC e la povera vittima inizierà a mandare tutti i pacchetti diretti al gateway a noi. Se poi agiamo

nello stesso modo sul gateway facendogli credere che all'IP della vittima corrisponda il nostro indirizzo MAC e ci preoccupiamo di reinviare ogni pacchetto che ci giunge al reale destinatario, la nostra impresa è compiuta.

A e B a livello logico non si accorgono di nulla e noi intanto leggiamo tutto quello che si dicono.

La pratica: spaventosamente semplice!

Ci sono diversi programmi che automatizzano quanto menzionato sopra; il più semplice di tutti è sicuramente *Ettercap*. Questo fenomenale programma dispone, oltre al classico metodo via riga di comando, anche di una modalità in shell interattiva o di una interfaccia GTK. Una volta selezionata l'interfaccia da utilizzare, basta fare una scansione degli host locali e avvelenare l'ARP con l'apposito comando, per vedersi sparare fuori tutti i pacchetti della rete.



HTTPS ci protegge... ma fino a che punto?

Per ovviare al problema dell'intercettazione dei pacchetti si è ben pensato di creare un protocollo sicuro. Infatti l'HTTPS (*HTTP² over Secure Sockets Layer*) consiste nel creare una connessione sicura tra client e server utilizzando un sistema di certificati e chiavi asimmetriche.

Una volta creato il canale sicuro si utilizza il normale protocollo HTTP. I pacchetti scambiati tramite questo protocollo intercettati risultano criptati e quindi incomprensibili. Questo protocollo è quello che solitamente viene usato sui siti di poste, banche, e più in generale di tutti quelle applicazioni online che hanno a che fare con soldi. Inoltre viene utilizzato anche da qualche famoso social network solo per il login.

Purtroppo esiste una geniale per quanto semplice tecnica chiamata 'SSLStrip'. In pratica l'idea è quella di far sì che tra la vittima e l'attaccante venga creata una normale connessione HTTP e poi è solo l'attaccante a creare la connessione HTTPS con il server che la richiede. In questo modo i pacchetti verranno criptati solo dopo essere arrivati all'attaccante.

Come difendersi?

Senza dubbio risulta utile fissare l'associazione IP-MAC cosicché l'indirizzo MAC

non possa essere cambiato da chiunque con un semplice pacchetto. Inoltre esistono dei programmi, come ad esempio *ARPalert*, che permette di eseguire un script quando viene cambiata una associazione IP-MAC. Potete quindi essere semplicemente avvisati con un messaggio sullo schermo oppure (per i più paranoici) potete far interrompere la connessione o spegnere il PC.

Per proteggersi dall'attacco SSLStrip basta fare attenzione al link del sito in questione, quindi evitare di dare numeri di carte se non si vede un *https* nel link.

Link Utili

Per approfondire: Ecco diversi collegamenti che possono tornare utili a chi vuole approfondire qualche questione trattata nell'articolo.

- **Ettercap:** <http://ettercap.sourceforge.net>
- **SSIStrip:** <http://www.thoughtcrime.org/software/sslstrip>
- **ArpAlert:** <http://www.arpalert.org>
- **HTTP:** http://it.wikipedia.org/wiki/Hypertext_Transfer_Protocol
- **MITM:** www.blackhats.it/en/papers/Paper-mitm.pdf

²*HyperText Transfer Protocol* (protocollo di trasferimento di un ipertesto) è usato come principale sistema per la trasmissione di informazioni sul Web.

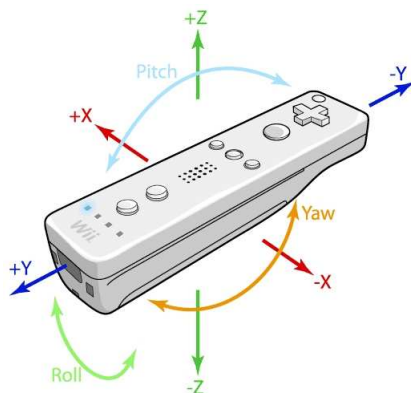
Utilizzare il Wiimote su Linux

Otacon22 <otacon22@email.it>

IL “WII REMOTE” o, come spesso viene chiamato su Internet, il “Wiimote”, è il controller ideato nel 2006 dalla Nintendo per l’innovativa console Wii.



me se non bastasse il tutto è interfacciato alla console di gioco senza fili mediante Bluetooth.



Questo semplice e accattivante controller ha conquistato il pubblico nel giro di pochi mesi, ed è probabilmente uno dei principali motivi di successo della piattaforma Wii. Il Wiimote possiede infatti, oltre ai classici pulsanti per muoversi nei giochi e l’effetto vibrazione, anche un accelerometro interno. Mediante questo la console di gioco è in grado di riconoscere l’accelerazione e l’inclinazione nello spazio del controller stesso. Nella parte frontale è presente anche una piccola telecamera a infrarossi che, abbinata alla cosiddetta “sensor bar”, permette alla console di sapere anche la nostra posizione rispetto allo schermo. Co-

Il fatto che sia Bluetooth significa però che è anche possibile interfacciare il Wiimote a qualunque altra cosa utilizzi il Bluetooth. Su Linux può sempre far comodo avere un controller come il Wiimote, come ad esempio quando stiamo giocando a qualche gioco “vintage” (sull’emulatore Xname, per esempio), quando vogliamo dilettarci a ruotare il nostro cubo di Compiz con un gesto della mano o per qualche pazzo esperimento.

Dopo esserci quindi dotati di un adattatore Bluetooth, su Internet troviamo diversi programmi (ma anche librerie) che ci permetteranno di interfacciare il Wiimote al

nostro PC ed utilizzarlo per muoverci nel nostro desktop o molto altro.

Wminput

Il programma probabilmente più comodo se vogliamo semplicemente spostare il mouse è *Wminput* (uno dei componenti di *Cwiid* di cui parleremo tra poco). Nella distribuzione Ubuntu possiamo già trovare il pacchetto nei repository ufficiali pronto da installare, che poi possiamo avviare da terminale:

```
$ sudo apt-get install wminput
$ sudo modprobe uinput
$ sudo wminput
```

Una volta lanciato il programma dovremo quindi premere contemporaneamente i pulsanti “1” e “2” sul nostro Wiimote per renderlo collegabile via Bluetooth. A questo punto il Wiimote si sarà effettivamente trasformato in un mouse: potremo muovere il cursore facendo ruotare il Wiimote su se stesso lateralmente o frontalmente.

Per cliccare con l’equivalente dei tasti sinistro e destro del mouse dovremo premere invece il tasto “A” e il tasto “B”. Possiamo anche scorrere le pagine con le frecce del wiimote e alzare il volume audio con i pulsanti “+” e “-”.

Wmgui

Anche questo è un componente della libreria *Cwiid*, che comunque troviamo in un pacchetto apposito su Ubuntu o altre distribuzioni. È utile per capire e testare

il funzionamento del nostro Wiimote: ci mostra tutti i dati di accelerazione, inclinazione, pulsanti e infrarossi. Dopo averlo installato e avviato con:

```
$ sudo apt-get install wmgui
$ wmgui
```

andiamo su *File > Connect* e, dopo aver premuto ancora “1” e “2” sul controller, premiamo *Ok* nel programma che si collegherà al Wiimote. Andando nel menu *Settings* abilitiamo la visione dei vari dati (anche eventualmente da un controller di estensione “nunchuk”).

Xwii

Non è ancora incluso all’interno dei pacchetti di Ubuntu e di molte altre distribuzioni, nonostante sia un comodo tool per utilizzare il nostro Wiimote come preferiamo.

Installiamo alcune dipendenze, scarichiamo il sorgente del programma e lo compiliamo:

```
$ sudo apt-get install libbluetooth-dev
libxtst-dev freeglut3-dev cl-sdl-opengl
build-essential
$ wget http://www.respect.com/xwii/files/
releases/2.8/XWii_2.8_source.tar.gz
$ tar xvf XWii_2.8_source.tar.gz
$ cd XWii_2.8_source/
$ make
```

Dopo aver finito di compilare possiamo lanciare il programma (sempre da terminale):

```
$ ./start_xwii.py
```

A questo punto ci comparirà un piccolo menu con una serie di configurazioni prefissate per utilizzare il Wiimote; scegliamo quella che ci interessa, come ad esempio “Tilt Mouse”, e digitiamo il numero corrispondente seguito da invio. Ora dobbiamo di nuovo abilitare il collegamento al wiimote premendo “1” e “2” e dopo qualche secondo potremo nuovamente usare il wiimote come mouse (come con *Wminput*). Se abbiamo a disposizione la “sensor bar” (che in realtà sono solo 2 led infrarossi e delle batterie) possiamo anche provare la modalità “IR Mouse”, in cui la posizione del cursore sarà determinata dalla posizione della “sensor bar” rispetto alla telecamera infrarossi del Wiimote.

E quale sarebbe il vantaggio di usare *Xwii*? Semplice: ora possiamo creare la nostra configurazione da caricare nel menu: spostiamoci nella cartella *profiles*, apriamo ad esempio il file *test.xwii* e osserviamolo. Tra le prime righe c'è ad esempio:

```
[home]KEYBOARD 1[/home]
```

Questa sintassi significa che alla pressione del tasto “home” sul controller dovrà essere associata sul computer l'azione della pressione sulla tastiera del tasto “1”. Potremmo quindi utilizzare ad esempio le frecce sul wiimote per giocare in un gioco FPS cambiando qualche riga della configurazione in questo modo:

```
[up]KEYBOARD w[/up]
[down]KEYBOARD s[/down]
[left]KEYBOARD a[/left]
[right]KEYBOARD d[/right]
```

Possiamo anche fare in modo che ad una forte accelerazione in qualunque direzione corrisponda una azione:

```
[flick]KEYBOARD Return[/flick]
```

Salviamo le modifiche e per testare il tutto riavvieremo il programma di prima, scegliendo nel menu la configurazione “test”.

Per consultare l'elenco delle “azioni” possibili può essere utile guardare su <http://abstrakraft.org/cwiid/wiki/wminput>

Wiican

Se siamo invece per una alternativa più comoda e grafica possiamo utilizzare *Wiican*, una GUI che riproduce le stesse funzioni di *Xwii*, con però una comoda icona nella traybar da cui impostare le varie configurazioni e profili. Non è purtroppo presente nei repository standard di Ubuntu, ma possiamo comunque installare il pacchetto facilmente:



```
$ sudo add-apt-repository # infrarossi rilevati dalla
ppa:fontanon/wiican      # telecamera IR
$ sudo apt-get update    wii.a.state["ir_src"]
$ sudo apt-get install wiican
```

Poi per avviarlo:

```
$ sudo wiican
```

Python-cwiid

Il Wiimote potrebbe però anche servirci per qualche pazzo esperimento, per questo motivo possiamo trovare su Internet la libreria *Cwiid* e in particolare il suo relativo porting in Python *python-cwiid* per interfacciarsi al controller. Possiamo trovare *python-cwiid* nei repository standard di Ubuntu e di varie altre distribuzioni. Per collegarsi al Wiimote sarà sufficiente qualche riga di codice; ecco qualche riga di esempio in Python:

```
import cwiid

# Dopo aver lanciato questa
# istruzione dovremo premere
# 1+2 sul controller
wii = cwiid.Wiimote()

# Accende il primo led sul
# controller
wii.led=1

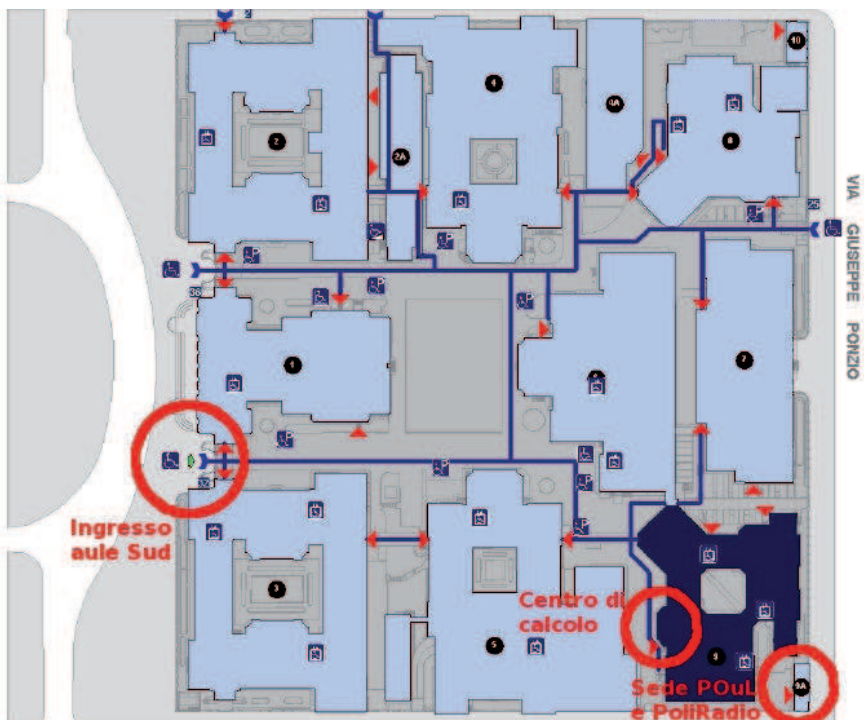
# Abilita la vibrazione
wii.rumble=True
wii.rpt_mode=8

# Restituisce una lista di
# dizionari con dati sui punti
```

Link Utili

Per approfondire:

- <http://it.wikipedia.org/wiki/Wiimote>
- <http://www.respect.com/xwii>
- <http://launchpad.net/wiican>
- <http://abstrakraft.org/cwiid>



Vi è venuta voglia di conoscere il mondo di Linux? Volete partecipare più da vicino alle nostre attività? Volete scrivere un articolo su questa rivista? Iscrivetevi alla nostra mailing list oppure venite a trovarci presso la nostra sede!

sito Internet: www.poul.org
informazioni: info@poul.org



La stampa della rivista è interamente finanziata dal Politecnico di Milano, che non si assume alcuna responsabilità sul contenuto.

Stampa a cura di *Acheias* di S. Siragusa, Milano 2009.