PANDA, una piattaforma per l'analisi dinamica del software

Giovanni Mascellani
gio@debian.org

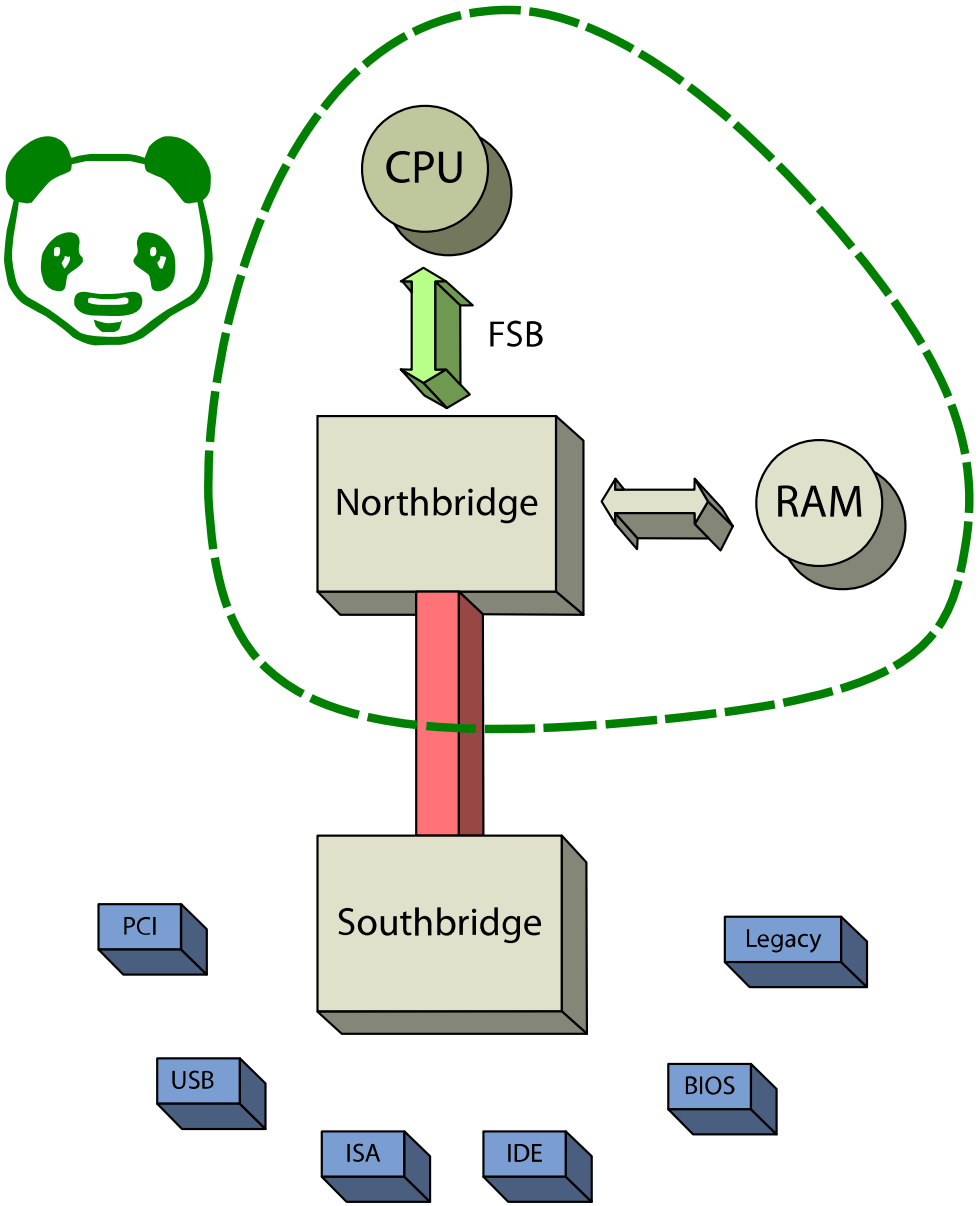
22 ottobre 2016
Linux Day – Gruppo Utenti Linux Pisa

Cosa è PANDA

- Framework per analisi dinamica del software; ossia, osservare il software mentre esegue (reverse engineering, analisi malware, …).
- Basato su QEMU 1.0.1 (aggiornamento WIP).
- Emula un intero sistema operativo e registra l’esecuzione.
- Poi si fa il replay e si analizza (plugin standard, plugin personalizzati, tool esterni).

Record & Replay

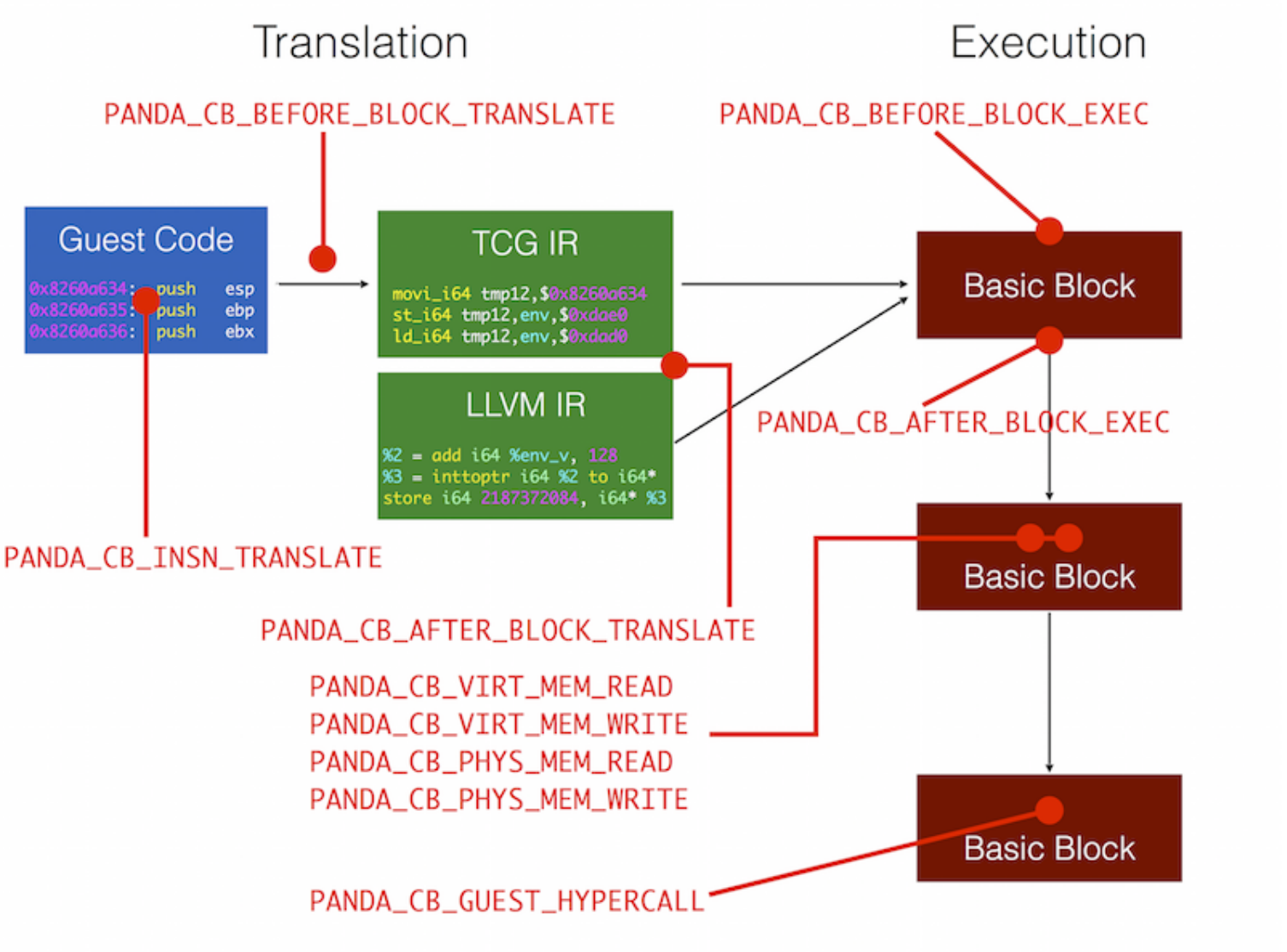
- Record: gli eventi non deterministici (IO, IRQ, MMIO) vengono registrati.
- Replay: a partire da uno snapshot iniziale ed il nondet log, PANDA può replicare esattamente l'esecuzione di una macchina virtuale.
- Durante il replay si emulano solo CPU e RAM, non le periferiche.
- Diverso da altri strumenti di R&R che registrano l'interazione di tutte le periferiche con il sistema esterno (indipendente dall'hardware emulato, no go live).



Plugin per l’analisi

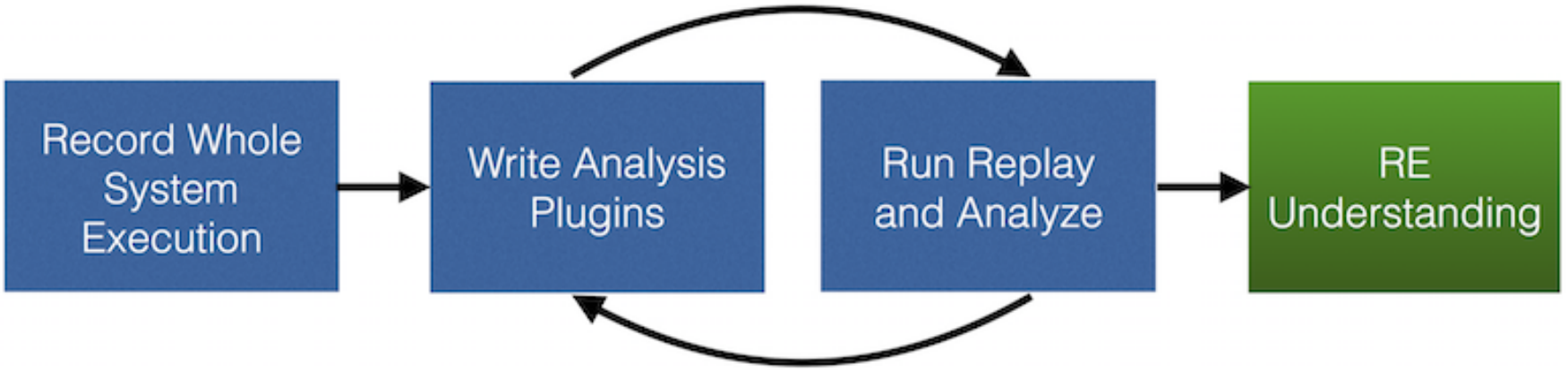
- Plugin standard: tainting, Tappan Zee, callstack tracking, OS introspection, misc...
- Facile scrivere nuovi plugin!
- Interazione con PANDA: callback (intercettazione RAM, …), code instrumentation (LLVM), accesso all’intero stato del sistema emulato.
- Interazione con altri plugin: callback, interfacce.

Callback per i plugin



Workflow

- Preparazione VM, esecuzione e registrazione.
- Eventuale distribuzione e condivisione su Internet. http://www.rrshare.org/
- Taglio della registrazione intorno ai punti critici.
- Analisi incrementale.



Vantaggi e limiti

- R&R (riproducibilità, parallelizzazione, condivisione, performance, archiviazione).
- Log R&R molto ridotti.
- Estensibilità.
- Whole system emulation.
- Trasparenza.
- Platform independence (es. Android).

- Basato su QEMU 1.0.1, che è vecchiotto (aggiornamento WIP).
- Non sfrutta KVM (fattibile, ma difficile).
- Trasparenza incompleta (malware difensivi).
- Hardware emulato scarso (no VGA accelerata).

Applicazioni

- Estrazione della black list di censura di un’app Android cinese di IM.
- RE di serial key.
- Diagnosi di vulnerabilità (use after free).
- Assistenza nella generazione automatica di bug (LAVA).

Miei esperimenti:

- Estrazione di chiavi segrete ECC.
- Riconoscimento di AES (per ora: del key schedule core).
- DEMO!

Ingredienti di AES

- S-box
- Rotazione di otto bit a sinistra: 1d2c3a4f → 2c3a4f1d
- unsigned char rcon[] = { 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36 }

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

AES key schedule core:

- Input: 4 byte
- Esegui una rotazione
- Applica S-box a ciascuno dei byte
- Xor del primo byte con rcon[i], dove i è il numero di iterazione
- Output: 4 byte

Bibliografia

- B. Dolan-Gavitt, J. Hodosh, P. Hulin, T. Leek, R. Whelan. *Repeatable Reverse Engineering with PANDA.* 5th Program Protection and Reverse Engineering Workshop, Los Angeles, California, December 2015.
- https://github.com/moyix/panda (con riferimenti agli articoli pubblicati, incluso Tappan Zee e LAVA)
- https://github.com/giomasce/panda (la mia branch con i plugin che ho sviluppato io)
- https://github.com/moyix/qemu (Panda 2 in sviluppo, “pretty close to ready”).
- http://www.rrshare.org/detail/50/ (il RR usato per la demo)