

Abbiamo le System Call che ci permettono di andare a SETTARE la regola di scheduling di uno specifico thread (noi passiamo un PID ma sappiamo benissimo che quando passiamo un PID identifichiamo anche un Thread ID), il secondo parametro identifica la policy con cui dobbiamo gestire questo thread, e poi passiamo un pointer ad una struttura dati che è sched_param, ossia i parametri di scheduling, e questa struttura dati al suo interno mantiene la scheduling_priority.

```
#include <sched.h>

int    sched_setscheduler(pid_t pid, int policy, const struct sched_param *p);
int    sched_getscheduler(pid_t pid);

struct sched_param {
    ...
    int sched_priority;
    ...
};
```

Possiamo andare a cambiare la gestione dello scheduling di questi thread dal punto di vista dell'assegnazione della CPU.

ABBIAMO ANCHE UNA GET CHE CI CONSEGNA LE INFORMAZIONI A LIVELLO DI PRIORITÀ ASSOCIATO AD UNO SPECIFICO THREAD CHE ABBIAMO ATTIVO ALL'INTERNO DEL SISTEMA.

Questi servizi possono essere utilizzati anche all'interno di comandi che noi possiamo lanciare all'interno di una shell.

Uno di questi interessanti è “Change Real Time”, ossia in CHRT.

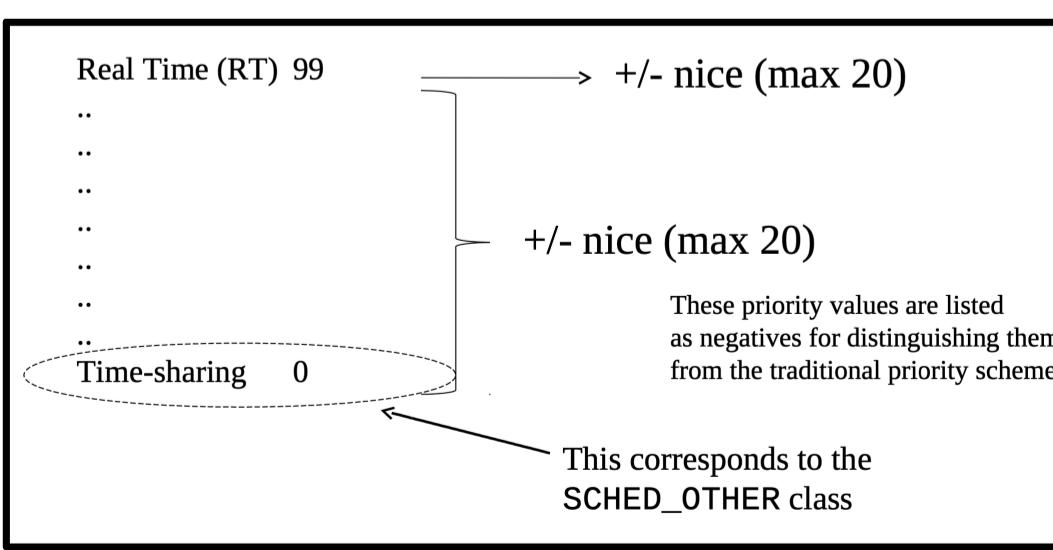
Questo comando ci permette di andare a controllare da shell il Linux Extended Priority Scheme, ossia uno schema di priorità dove ci basiamo su concetti relativi al Real Time. Con questi comandi cambiamo il livello di priorità all'interno di questo schema esteso.

Ma come è fatto questo schema esteso?

Actual priorities with the extended scheme

Esistono 100 classi di priorità differenti.

All'interno dello schema, la classe zeroesima rappresenta tutto ciò che abbiamo già spiegato per quanto riguarda lo scheduler unix tradizionale: se un thread è correttamente inserito nella classe Time-Sharing, quel thread appartiene esattamente ad una di quelle 40 possibilità che abbiamo già introdotto.



All'interno di ciascuna classe possiamo avere una NICE, cioè all'interno di questa classe possiamo essere più o meno importanti

UNICO IMPORTANTE

La time-sharing ingloba esattamente l'espressione delle classi di priorità che noi avevamo all'interno dello scheduler tradizionale.

Le classi di attività superiore sono orientate nel gestire attività real time, non time-sharing: quando noi abbiamo un thread che non è registrato a livello 0, in realtà questo thread ha la sua priorità di riferimento che non cambia in funzione di quanta CPU quel thread ha utilizzato, come magari accadeva principalmente per la classe 0. Quindi ciò che è sotto nella classe 0 viene mantenuto così come abbiamo lavorato nello UNIX tradizionale, tutto ciò che è sopra lavora esclusivamente utilizzando la priorità di riferimento, che però, NON È PIÙ FUNZIONE DEL TEMPO IN CUI UN THREAD OCCUPA LA CPU, quindi la priorità di riferimento è sicuramente in funzione della NICE e della classe di priorità che ci va a definire un concetto di base. Ma non abbiamo più una variazione per quanto riguarda l'utilizzo della CPU.

Questo è un modo più semplice per controllare applicazioni che sono orientate al real-time, perché se io so che un'applicazione è importante e questa applicazione all'interno della classe che io gli assegno non deve essere declassata in funzione del tempo di CPU che ha utilizzato, questa applicazione può non prendere la CPU non perché ha utilizzato la CPU precedentemente, ma può non prendere la CPU semplicemente perché ho qualcosa di più importante da fare. Qui si parla di soft real time.

Se sulla SHELL listiamo le applicazioni con TOP e vediamo la lista delle applicazioni con le relative priorità possiamo osservare la seguente cosa:

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
3951	frances+	20	0	3968300	796236	97704	S	86.05	4.927	39:40.14	teams
3778	frances+	20	0	5080724	309456	227696	S	36.54	1.915	53:35.49	VirtualBox
4267	frances+	20	0	22.919g	357012	155564	S	33.55	2.209	20:08.54	chrome
2374	frances+	9	-11	2484576	22248	17156	S	12.62	0.138	7:37.89	pulseaudio
1989	root	20	0	368160	124580	83384	S	9.967	0.771	6:56.19	X
2815	frances+	20	0	894060	92652	56332	S	8.638	0.573	5:37.64	teams
2326	frances+	20	0	3240260	109740	68208	R	5.980	0.679	4:15.84	kwin_x11
1138	root	-51	0	0	0	0	D	5.648	0.000	2:22.71	irq/133-rmt4_sm
3917	frances+	20	0	2777852	673560	84956	S	5.648	4.168	4:38.65	teams
2619	frances+	20	0	3261760	235308	105316	S	3.987	1.456	3:32.45	teams
4113	frances+	20	0	641704	184400	144888	S	3.654	1.141	2:19.45	chrome
4323	frances+	20	0	1040568	65456	53360	S	2.658	0.405	1:32.86	chrome
3468	frances+	20	0	682496	76348	61632	S	1.661	0.472	0:01.84	konsole
4079	frances+	20	0	836740	199248	130040	S	0.997	1.233	0:36.37	chrome
4115	frances+	20	0	525552	84180	65184	S	0.664	0.521	0:22.53	chrome
2672	frances+	20	0	3234328	299000	76084	S	0.332	1.850	0:17.59	dropbox
5038	root	20	0	0	0	0	S	0.332	0.000	0:00.64	kworker/u8:2
5806	frances+	20	0	41356	3884	3268	R	0.332	0.024	0:00.04	top
1	root	20	0	157012	9276	6852	S	0.000	0.057	0:03.09	systemd
2	root	20	0	0	0	0	S	0.000	0.000	0:00.00	kthread
4	root	0	-20	0	0	0	S	0.000	0.000	0:00.00	kworker/0:0H
6	root	0	-20	0	0	0	S	0.000	0.000	0:00.00	mm_percpu_wq
7	root	20	0	0	0	0	S	0.000	0.000	0:00.84	ksoftirqd/0
8	root	20	0	0	0	0	S	0.000	0.000	0:04.12	rcu_sched
9	root	20	0	0	0	0	S	0.000	0.000	0:00.00	rcu_bh
10	root	rt	0	0	0	0	S	0.000	0.000	0:00.00	migrator/0
11	root	rt	0	0	0	0	S	0.000	0.000	0:00.00	watchdog/0
12	root	20	0	0	0	0	S	0.000	0.000	0:00.00	cphuph/0

In particolare al livello 50 di questa tabella. Qui viene scritto come valore negativo e incrementato di una unità semplicemente per come è strutturata l'applicazione TOP, ma stiamo dicendo che c'è qualche thread assegnato alla classe di priorità con TAG 50. La sua nice è zero però (COLONNA NI).

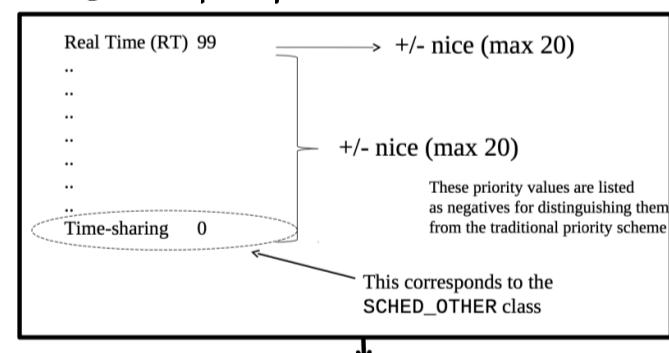
Possiamo avere anche scenari in cui a -51 noi possiamo avere una Nicesse diversa da zero, ciò significa che potevamo assegnare una priorità superiore o inferiore all'interno di quella specifica classe. Chiaramente quando abbiamo -51 abbiamo qualcosa di significativamente più importante rispetto a ciò che rappresentiamo con valori positivi come 20, che all'interno dello schema sono tutte applicazioni dentro la classe Time-Sharing.

Ad esempio con Nicesse -20 portiamo al livello 0, quindi al livello più alto all'interno di quella classe di priorità, l'applicazione di cui stiamo parlando.

Questo ci permette di avere un carico time-sharing dove le decisioni di assegnazione della CPU sono prese al fine di ottimizzare il comportamento del sistema. Però se abbiamo applicazioni più importanti che vanno verso il real time, diamo la cpu a queste cose più importanti (-51).

La priorità in taluni casi poteva essere marcata con un valore positivo, per esempio la 20, in taluni casi con un valore negativo, -51.

Quando leggiamo un valore negativo significa che il thread con il PID 1138 associato al livello di priorità -51, è esattamente uno dei thread che sta eseguendo qua sopra.



PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5915	frances+	20	0	4144	844	784	R	94.04	0.005	0:20.45	a.out
3951	frances+	20	0	4051252	884868	97704	S	69.54	5.476	42:29.93	teams
4267	frances+	20	0	22.919g	367840	155496	S	39.07	2.276	21:37.61	chrome
3778	frances+	20	0	5080724	309456	227696	S	38.08	1.915	55:53.29	VirtualBox
2374	frances+	9	-11	2484576	22248	17156	S	12.58	0.138	8:11.28	pulseaudio
1989	root	20	0	359984	124108	82912	S	10.93	0.768	7:23.70	X
2815	frances+	20	0	306408	92024	56526	S	9.600	0.576	6:02.97	teams

Il primo a.out è partito usando la classe di priorità tradizionale, in particolare la time sharing, ma possiamo ovviamente andare

2813 frances+	20	0	690406	93024	50550	S	8.009	0.570	0:02.67	teams
2326 frances+	20	0	3278596	120076	73132	S	6.291	0.743	4:32.54	kwin_x11
1138 root	-51	0	0	0	0	D	5.298	0.000	2:34.95	irq/133-rmi4_sm
3917 frances+	20	0	2780924	676396	85008	S	4.636	4.186	4:56.52	teams
2619 frances+	20	0	3263872	235428	105364	S	4.305	1.457	3:45.89	teams
4113 frances+	20	0	641736	184404	144892	S	3.642	1.141	2:29.70	chrome
4323 frances+	20	0	1040568	65456	53360	S	2.318	0.405	1:39.84	chrome
3468 frances+	20	0	682496	76348	61632	S	1.325	0.472	0:03.86	konsole
4079 frances+	20	0	836740	199820	129944	S	0.993	1.237	0:39.11	chrome
410 root	-2	0	0	0	0	S	0.331	0.000	0:03.25	i915/signal:0
526 root	20	0	12140	7424	1620	S	0.331	0.046	0:06.03	havaged
1466 nscd	20	0	939216	2876	2332	S	0.331	0.018	0:00.26	nscd
2492 frances+	20	0	620268	43824	39152	S	0.331	0.271	0:00.50	akonadi_contact
2494 frances+	20	0	634560	45844	40960	S	0.331	0.284	0:00.51	akonadi_ical_re
2503 frances+	20	0	716864	46528	41360	S	0.331	0.288	0:00.71	akonadi_maildis
2523 frances+	20	0	1128540	82416	66432	S	0.331	0.510	0:00.82	akonadi_sendlat
2672 frances+	20	0	3234328	301264	76084	S	0.331	1.864	0:18.15	dropbox
4115 frances+	20	0	526576	84220	65184	S	0.331	0.521	0:24.10	chrome

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+ COMMAND
5915 frances+	-2	0	4144	844	784	R	100.0	0.005	1:08.44	a.out
3951 frances+	20	0	4050740	884892	97704	S	67.66	5.476	43:17.79	teams
4267 frances+	20	0	22.917g	361748	155496	S	34.98	2.239	21:54.02	chrome
3778 frances+	20	0	5080724	309456	227696	S	32.01	1.915	56:10.41	VirtualBox
2374 frances+	9	-11	2484576	22248	17156	S	12.54	0.138	8:17.25	pulseaudio
1989 root	20	0	360304	124640	83444	S	11.88	0.771	7:28.84	X
1138 root	-51	0	0	0	0	D	9.241	0.000	2:36.71	irq/133-rmi4_sm
2815 frances+	20	0	896408	93024	56536	S	8.581	0.576	6:06.95	teams
2326 frances+	20	0	3278594	120076	73132	S	6.601	0.743	4:35.43	kwin_x11
3917 frances+	20	0	2780924	676924	85064	S	4.950	4.179	4:59.04	teams
2619 frances+	20	0	3263872	235428	105364	S	4.290	1.457	3:48.17	teams
4113 frances+	20	0	641736	184404	144892	S	3.300	1.141	2:31.36	chrome
4323 frances+	20	0	1040568	65456	53360	S	2.310	0.405	1:41.03	chrome
3468 frances+	20	0	693720	78092	63284	S	1.650	0.483	0:04.51	konsole
4079 frances+	20	0	836740	199828	129944	S	0.660	1.237	0:39.47	chrome
14 root	rt	0	0	0	0	S	0.330	0.000	0:00.01	watchdog/1
2311 frances+	20	0	534644	39748	34456	S	0.330	0.246	0:00.49	kglobalaccel5
2344 frances+	20	0	372524	21124	19052	S	0.330	0.131	0:00.39	xembedsniproxy
2379 frances+	20	0	795144	33156	26480	S	0.330	0.205	0:00.54	kactivitymanager
2490 frances+	20	0	1133100	84604	68004	S	0.330	0.524	0:00.64	akonadi_archive
2493 frances+	20	0	717620	48792	41588	S	0.330	0.302	0:00.73	akonadi_followu
2672 frances+	20	0	3234328	301536	76084	S	0.330	1.866	0:18.54	dropbox
3753 frances+	20	0	821924	23412	18736	S	0.330	0.145	0:06.61	VBoxSVC
4115 frances+	20	0	526576	84220	65184	S	0.330	0.521	0:24.37	chrome

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+ COMMAND
5915 frances+	-2	1	4144	844	784	R	100.0	0.005	1:32.34	a.out
3951 frances+	20	0	4050740	884872	97704	S	50.00	5.476	43:16.76	teams
4267 frances+	20	0	22.918g	356948	155496	S	37.50	2.209	22:02.02	chrome
3778 frances+	20	0	5080724	309456	227696	S	31.25	1.915	56:18.60	VirtualBox
1989 root	20	0	360272	124644	83448	R	12.50	0.771	7:31.35	X
2374 frances+	9	-11	2484576	22248	17156	S	12.50	0.138	8:20.14	pulseaudio
5968 frances+	20	0	41388	4060	3448	R	12.50	0.025	0:00.02	top
2619 frances+	20	0	3263872	235428	105364	R	6.250	1.457	3:49.23	teams
2815 frances+	20	0	896408	93024	56536	S	6.250	0.576	6:08.91	teams
4323 frances+	20	0	1040568	65456	53360	S	6.250	0.405	1:41.61	chrome
1 root	20	0	157012	9276	6852	S	6.000	0.057	0:03.09	systemd
2 root	20	0	0	0	0	S	0.000	0.000	0:00.00	kthread
4 root	0	-20	0	0	0	S	0.000	0.000	0:00.00	kworker/0:H
6 root	0	-20	0	0	0	S	0.000	0.000	0:00.00	mm_percpu_wq
7 root	20	0	0	0	0	S	0.000	0.000	0:00.92	ksoftirqd/0
8 root	20	0	0	0	0	S	0.000	0.000	0:04.45	rcu_sched
9 root	20	0	0	0	0	S	0.000	0.000	0:00.00	rcu_bh
10 root	rt	0	0	0	0	S	0.000	0.000	0:00.00	migration/0
11 root	rt	0	0	0	0	S	0.000	0.000	0:00.00	watchdog/0
12 root	20	0	0	0	0	S	0.000	0.000	0:00.00	cpuhp/0
13 root	20	0	0	0	0	S	0.000	0.000	0:00.00	cpuhp/1
14 root	rt	0	0	0	0	S	0.000	0.000	0:00.01	watchdog/1
15 root	rt	0	0	0	0	S	0.000	0.000	0:00.01	migration/1
16 root	20	0	0	0	0	S	0.000	0.000	0:00.09	ksoftirqd/1

Se siamo nei livelli real time, espressi come valori negativi (livello 50 = -50-1 = -51), e abbiamo due livelli di priorità 100 e uno con 101 indipendentemente dalla nice, passa prima quello con priorità 10