# Procesy a úlohy

| **PROCESY** | |
|---|---|
| **chkconfig** [<volba> [<sluzba> [<dostupnost>]]] | **--add** <sluzba> přidá službu pod správu chkconfig, **--del** <sluzba> odstraní službu, **--level** <uroven_behu_systemu> určí úroveň běhu systému, **--list** / **--list** <sluzba> vypíše všechny služby / danou službu a jejich/její dostupnost {off \| on} na všech úrovních běhu systému<br>`# chkconfig httpd on`<br>(povolí službu na úrovních běhu systému 2, 3, 4 a 5)<br>`# chkconfig --level 234 qemu off`<br>(zakáže službu na úrovních běhu systému 2, 3 a 4)<br>`$ chkconfig --list \| egrep -i "rexec\|rlogin\|rsh"`<br>(vypíše dané služby a jejich dostupnost na všech úrovních běhu systému) |
| **systemctl** <prikaz> [<sluzba ...>]<br>(platí od RHEL 7) | **enable** povolí službu, **disable** zakáže službu, **is-enabled** zjistí dostupnost služby („service unit“) automaticky po startu systému, **mask** zabrání automatickému či manuálnímu spuštění služby, **unmask** zruší účinek příkazu „mask“, **--user** spravuje uživatelské systemd soubory (služby)<br>`# systemctl enable httpd`<br>(povolí službu automaticky po startu systému)<br>`# systemctl mask postfix.service`<br>(zabrání automatickému či manuálnímu spuštění služby)<br>`$ systemctl list-unit-files -t service`<br>(vypíše všechny služby a jejich dostupnost) |
| **service** [<sluzba>] <prikaz> | **start** spustí službu, **stop** zastaví službu, **restart** restartuje službu, **reload** znovu načte konfigurační soubor služby, **status** zobrazí stav služby<br>`# service cups restart \| # /etc/init.d/cups restart`<br>(restartuje danou službu)<br>`# service --status-all`<br>(zobrazí stav všech služeb) |
| **systemctl** <prikaz> [<sluzba ...>]<br>(platí od RHEL 7) | **start** spustí službu, **stop** zastaví službu, **restart** restartuje službu, **reload** znovu načte konfigurační soubor služby, **status** zobrazí stav služby („service unit“), **is-active** vypíše, zda je služba aktivní (běží) nebo neaktivní, **--user** spravuje uživatelské systemd soubory (služby)<br>`# systemctl restart cups`<br>(restartuje danou službu)<br>`# systemctl list-units -t service`<br>(zobrazí stav nainstalovaných služeb)<br>`# systemctl list-units -t service --state=running`<br>(zobrazí běžící služby)<br>`# systemctl list-units -t service --all`<br>(zobrazí stav všech služeb) |
| **systemctl list-jobs**<br>(platí od RHEL 7) | vypíše všechny úlohy, které systemd aktuálně provádí |

| **PROCESY** | |
|---|---|
| <prikaz> [<argument ...>] | spustí spustitelný program (binární soubor) v aktuálním shellu; program je vestavěný příkaz shellu či příkaz nalezený v $PATH nebo v zadané cestě<br>`$ date` |
| **bash** \| **sh** [<prikaz \| soubor> [<argument ...>]] | spustí shell, příkaz či soubor (i bez nastavených práv ke spuštění) ve vnořeném shellu; soubor hledá v pracovním adresáři a v $PATH, **-r** spustí omezený shell, **-c** <retezec> čte příkazy z daného řetězce, **-s** čte příkazy ze STDIN, **-x** spustí skript v debug režimu<br>`$ sudo bash -c "echo 'Vitejte na moji strance' > /var/www/html/index.html"`<br>(přesměruje výstup příkazu do daného souboru bez nutnosti přihlásit se na účet uživatele root) |
| **source** \| **.** <soubor> [<argument ...>] | spustí daný soubor (i bez nastavených práv ke spuštění) v aktuálním shellu; soubor hledá v pracovním adresáři a v $PATH<br>`$ . ~/.bash_profile`<br>(obnoví profil uživatele bez nutnosti opětovného přihlášení) |
| <soubor> [<argument ...>] | spustí spustitelný soubor ve vnořeném shellu; soubor hledá v $PATH nebo v zadané cestě<br>`$ ./skript` |
| **command** <prikaz> [<argument ...>] | spustí vestavěný příkaz shellu nebo příkaz nalezený v $PATH, přičemž ignoruje jakoukoliv funkci shellu pojmenovanou shodně s názvem příkazu (shell hledá název příkazu v tomto pořadí: funkce shellu, vestavěné příkazy shellu, spustitelné soubory v $PATH) |
| **builtin** <prikaz> [<argument ...>] | spustí vestavěný příkaz shellu, přičemž ignoruje jakoukoliv funkci shellu pojmenovanou shodně s názvem příkazu (shell hledá název příkazu v tomto pořadí: funkce shellu, vestavěné příkazy shellu, spustitelné soubory v $PATH) |
| **exec** [<prikaz> [<argument ...>]] [<presmerovani ...>] | nahradí shell daným příkazem (nevytvoří se žádný nový proces) nebo provede přesměrování<br>`$ exec bash`<br>(nahradí shell bashem)<br>`$ exec > output.txt`<br>(přesměruje veškerý výstup do souboru „output.txt“ pro aktuální proces shellu) |
| **pstree** [<PID \| uzivatel>] | zobrazí stromovou strukturu procesů v abecedním pořadí, počínaje procesem "init" (systemd) nebo daným PID či procesy vlastněnými daným uživatelem, **-n** v číselném pořadí (podle PID), **-p** vypíše PID, **-u** vypíše uživatele |

| **PROCESY** | |
|---|---|
| **ps** | zobrazí statický seznam běžících procesů přihlášeného uživatele na daném terminálu (PID – identifikační číslo procesu, TTY – terminál, ze kterého byl proces spuštěn (znak „?“ znamená, že s procesem není spojen žádný terminál), TIME – čas, po který byl proces zpracováván procesorem, CMD – příkaz či proces a jeho parametry), **-e** všechny procesy všech uživatelů, **-f** podrobný výpis (navíc zobrazí UID – vlastníka procesu, PPID – rodičovský proces daného procesu, C – využití procesoru, STIME – čas spuštění procesu), **-l** úplný výpis (navíc zobrazí F – příznak procesu (0 – bežný proces, 1 – rozvětvený, ale nespuštěný proces, 4 – proces běžící s právy roota, 5 – kombinace předchozích hodnot), S – stav procesu („R“ běžící proces, „S“ spící proces (čekající na dokončení události), „D“ spící proces nereagující na signály (obvykle I/O), „T“ pozastavený proces, „Z“ zombie (defunct) proces – ukončený proces, který stále zabírá PID, jelikož rodičovský proces si není vědom jeho stavu ukončení), C – vytížení procesoru v %, PRI – prioritu procesu, NI – hodnotu priority zadanou příkazem „nice“, SZ – celkovou velikost paměti v blocích, kterou proces zabírá, WCHAN – adresu funkce jádra, kde proces čeká na určitou událost (u běžícího procesu je „-“)), **-p** <PID> daný proces, **-C** <program> procesy daného programu, **-U** <uzivatel \| UID> procesy daného uživatele, **-G** <skupina \| GID> procesy dané skupiny, **-t** <terminal> procesy na daném terminálu, **-o** <format> podle daného formátu, **--sort** <specifikace> třídí procesy podle dané specifikace, **-Z** vypíše bezpečnostní kontext SELinuxu<br>`$ ps -ef \| grep sshd`<br>(vyhledá v běžících procesech daný proces)<br>`$ ps -lt pts/1`<br>(vypíše podrobný seznam běžících procesů na daném terminálu)<br>`$ ps -p $$`<br>(vypíše aktuální proces shellu)<br>`$ ps -p 1 -o comm=`<br>(vypíše jméno procesu podle jeho PID)<br>`$ ps -eo pid,ppid,user,%mem,%cpu,comm --sort=-%cpu`<br>(třídí procesy podle vytížení procesoru)<br>`$ ps -eo pid,ppid,user,%mem,%cpu,comm --sort=-%mem`<br>(třídí procesy podle vytížení paměti) |
| **top** | zobrazí interaktivním a dynamickým způsobem seznam procesů, včetně jejich PID, vlastníka, priority nebo stavu a celkové využití CPU, paměti a swapu, **-u** <uzivatel> procesy daného uživatele, **-p** <PID> daný proces; interaktivní volby: **f** zobrazí nabídku položek, které lze přidat jako další sloupec do výstupu příkazu (např. swap, UID či PPID), **M** seřadí procesy podle využití paměti, **P** seřadí procesy podle využití procesoru (implicitně), **k** <PID> ukončí daný proces, **q** ukončí program |

| **PROCESY** | |
|---|---|
| **pidof** <jmeno_procesu> | zobrazí PID daného procesu<br>`$ ps -p $(pidof sshd)`<br>(vyhledá v běžících procesech daný proces) |
| **pgrep** <vzor> | vypíše PID procesů odpovídajících vzoru, **-f** odpovídá celému příkazovému řádku (nejen názvu procesu), včetně cesty ke spustitelnému souboru, argumentů a voleb, **-l** vypíše i jméno procesu, **-P** <PPID> vypíše procesy nadřazeného procesu, **-c** vypíše počet odpovídajících procesů, **-i** ignoruje velikost písmen, **-u** <uzivatel \| UID> procesy daného uživatele<br>`$ pgrep http`<br>(vypíše PID httpd procesů) |
| **nice** [[[**-n**] <priorita>] <prikaz>] | spustí proces se změněnou prioritou (hodnotou „nice“), parametr „priorita“ uvádí hodnotu, která se odečte nebo přičte k výchozí hodnotě (implicitně 0), pokud se vynechá, hodnota se automaticky sníží o 10; priorita se udává v rozmezí -20 až 19, čím nižší číslo, tím vyšší priorita, záporné hodnoty smí nastavit pouze root; bez argumentu vypíše aktuální zděděnou prioritu<br>`$ nice -n 15 rm -rf ~/tmp/*`<br>(sníží prioritu procesu o 15)<br>`# nice -n -15 ls /etc \| cpio -ov > /dev/rmt0`<br>(zvýší prioritu procesu o 15) |
| **renice** [**-n**] <priorita> <identifikator> | změní prioritu (hodnotu „nice“) běžícího procesu, hodnota priority se zadává v absolutním tvaru v rozmezí -20 až 19; běžný uživatel smí prioritu pouze snižovat, [**-p**] <PID> určí proces, **-u** <uzivatel> určí procesy daného uživatele, **-g** <skupina> určí procesy dané skupiny<br>`# renice -n 5 987`<br>(nastaví prioritu daného procesu)<br>`# renice -n -20 -u root`<br>(zvýší prioritu všech procesů uživatele root na maximum)<br>`# renice -n 19 -g users`<br>(sníží prioritu všech procesů skupiny „users“ na minimum) |
| **Ctrl**+**c** | ukončí běžící proces v popředí |
| **Ctrl**+**d** | ukončí běžící proces, který čte data ze STDIN nebo ukončí současný shell |

| **PROCESY** | |
|---|---|
| **kill** [<signal>] <PID ... \| **%**cislo_ulohy ...> | (**-15**) ukončí proces či úlohu odpovídající danému ID standardním způsobem (signál SIGTERM), **-9** ukončí proces ihned (signál SIGKILL), **-1** ukončí proces, u démonů znovu načte jejich konfigurační soubor (signál SIGHUP), **-17** upozorní rodičovský proces na události podřízených procesů (signál SIGCHLD), **-19** pozastaví proces (signál SIGSTOP), **-18** spustí pozastavený proces (signál SIGCONT), **-l** vypíše seznam všech signálů, **-s** <signal> určí signál; signál se zadává číslem nebo jménem (s předponou „SIG“ či bez ní a bez ohledu na velikost písmen)<br>`# kill -9 1`\|`# kill -SIGKILL 1`\|`# kill -KILL 1`\|`# kill -sigkill 1`\|`# kill -kill 1`<br>(ukončí ihned daný proces)<br>`$ kill %3`<br>(ukončí danou úlohu)<br>`$ kill -s SIGCHLD 1376`<br>(ukončí zombie proces odesláním signálu SIGCHLD rodičovskému procesu) |
| **killall** [<signal>] <jmeno_procesu ...> | (**-15**) ukončí všechny procesy odpovídající danému jménu (vhodné zvlášť při více spuštěných procesech jednoho programu) standardním způsobem (signál SIGTERM), **-9** ukončí proces ihned (signál SIGKILL), **-19** pozastaví proces (signál SIGSTOP), **-18** spustí pozastavený proces (signál SIGCONT), **-I** ignoruje velikost písmen, **-l** vypíše seznam všech signálů, **-s** <signal> určí signál, **-u** <uzivatel> ukončí všechny procesy daného uživatele; signál se zadává číslem nebo jménem (s předponou „SIG“ či bez ní a bez ohledu na velikost písmen)<br>`# killall -9 sshd`<br>(ukončí ihned dané procesy) |
| **pkill** [<signal>] <vzor> | (**-15**) ukončí proces odpovídající danému vzoru standardním způsobem (signál SIGTERM), **-9** ukončí proces ihned (signál SIGKILL), **-19** pozastaví proces (signál SIGSTOP), **-18** spustí pozastavený proces (signál SIGCONT), **-f** odpovídá celému příkazovému řádku (nejen názvu procesu), včetně cesty ke spustitelnému souboru, argumentů a voleb, **-i** ignoruje velikost písmen, **-u** <uzivatel \| UID> procesy daného uživatele; signál se zadává číslem nebo jménem (s předponou „SIG“ či bez ní a bez ohledu na velikost písmen) |
| **fuser** [<soubor ... \| zarizeni ...>] | vypíše procesy používající daný soubor či zařízení, **-m** vypíše procesy používající celý souborový systém, na kterém daný soubor leží, **-u** vypíše vlastníka procesu, **-k** ukončí proces, **-i** žádá o potvrzení, **-v** podrobný výpis<br>`$ fuser -kiv /dev/cdrom`<br>(ukončí procesy používající dané zařízení) |
| **strace** [<prikaz>] | sleduje systémová volání a signály volané a přijímané procesem, **-e** <vyraz> sleduje dané systémové volání, **-o** <soubor> zapíše výstup do souboru, **-p** <PID> sleduje běžící proces, **-c** počítá čas, volání a chyby pro každé systémové volání a zobrazí shrnutí při ukončení programu |

| **PROCESY** | |
|---|---|
| **ltrace** [<prikaz>] | sleduje volání sdílených knihoven a signály volané a přijímané procesem, **-e** <vyraz> sleduje dané volání knihovny, **-o** <soubor> zapíše výstup do souboru, **-p** <PID> sleduje běžící proces, **-c** počítá čas a volání pro každé volání knihovny a zobrazí shrnutí při ukončení programu, **-S** navíc zobrazí systémová volání |
| **rpcinfo** [<IP_adresa \| jmeno_pocitace>] | **-p** vypíše RPC služby (jejich identifikační číslo, verzi, protokol, port a název) běžící pod správou portmapperu na lokálním či daném počítači; seznam všech RPC programů je uveden v */etc/rpc* |
| **clustat** | zobrazí stav clusteru |
| **pcs status**<br>(platí od RHEL 6) | zobrazí stav clusteru |

| **ÚLOHY** | |
|---|---|
| **jobs** [**%**<cislo_ulohy> ...] | vypíše danou úlohu (její číslo, „+“ či „-“ označující poslední nebo předchozí úlohu, status a příkaz spojený s úlohou) v aktuálním shellu, **-l** včetně PID, **-r** vypíše pouze běžící úlohy, **-s** vypíše pouze pozastavené úlohy, **-n** vypíše pouze změny od předchozího spuštění; bez argumentu vypíše všechny pozastavené úlohy a úlohy běžící na pozadí |
| <prikaz> **&** | spustí úlohu na pozadí ve vnořeném shellu |
| **Ctrl**+**z** | pozastaví běžící úlohu v popředí |
| **suspend** | pozastaví provádění aktuálního shellu, dokud neobdrží signál SIGCONT, **-f** pozastaví přihlašovací shell |
| **bg** [**%**<cislo_ulohy> ...] | spustí pozastavenou úlohu na pozadí; bez argumentu spustí poslední pozastavenou úlohu |
| **fg** [**%**<cislo_ulohy>] | spustí pozastavenou úlohu v popředí nebo úlohu z pozadí přesune do popředí; bez argumentu spustí poslední pozastavenou úlohu či přesune poslední spuštěnou úlohu na pozadí do popředí |
| **nohup** <prikaz> | spustí daný příkaz tak, že je odolný vůči signálu SIGHUP (běží i po odhlášení uživatele); výstup příkazu se ukládá do souboru „nohup.out“ v pracovním adresáři<br>`# nohup find /tmp -name core -print > core.txt &`<br>(spustí na pozadí daný příkaz, který běží i po odhlášení uživatele) |
| **watch** <prikaz> | spustí příkaz opakovaně v pravidelných intervalech 2 sekund, **-n** <n> každých *n* sekund, **-d** zvýrazní změny na výstupu, **Ctrl**+**c** ukončí program<br>`$ watch -n 1 date`<br>(zobrazí aktuální čas)<br>`$ watch -d iostat`<br>(zobrazí změny na I/O zatížení disků a rychlosti zápisu a čtení dat)<br>`$ watch "ps -eo %cpu,pid,user,args --sort=-%cpu \| head -11"`<br>(zobrazí dynamický seznam 10 procesů, které nejvíce využívají procesor) |

| **ÚLOHY** | |
|---|---|
| **wait** [<PID \| **%**cislo_ulohy>] | čeká na ukončení daného procesu či úlohy běžící na pozadí aktuálního shellu, pak zanikne; vrátí návratový kód posledního procesu, pokud proces neexistuje, vrátí kód 127; bez argumentu čeká na ukončení všech procesů běžících na pozadí aktuálního shellu a návratový kód je 0<br>`$ wait $! && echo "OK"`<br>(čeká na ukončení posledního procesu běžícího na pozadí) |
| **sleep** <doba> | nastaví časový interval, během něhož nedochází k žádné aktivitě (vhodné zejména v případě, kdy je třeba zajistit určitou časovou prodlevu před spuštěním dalšího příkazu); doba trvání je uvedena číslem a příponou **s** v sekundách (implicitně), **m** v minutách, **h** v hodinách, **d** ve dnech<br>`$ sleep 10 && echo "Uplynulo 10 sekund."`<br>(spustí následující příkaz po uplynutí 10 sekund) |
| **at** <cas> [<datum>] [<dalsi_casove_urceni>] | spustí úlohu jednorázově v uvedený čas; příkazy, které se mají provést, se čtou ze STDIN nebo ze souboru, **-f** <soubor> čte příkazy z daného souboru, **-c** <cislo_ulohy> zobrazí obsah naplánované úlohy, **-d** <cislo_ulohy> odstraní naplánovanou úlohu (obdoba příkazu „atrm“), **-l** vypíše seznam naplánovaných úloh (obdoba příkazu „atq“), **-m** odešle uživateli email po dokončení úlohy<br>`$ at 20:00 25.06.2007 <-'` \| `$ at 7am today + 3 weeks <-'`<br>`at> mail root < nabidka.txt <-'`<br>`at>` **Ctrl**+**d**<br>(odešle v daný čas uživateli root email s obsahem uvedeného souboru)<br>`$ echo "date > date.txt" \| at noon Sunday`<br>(spustí v daný čas příkaz, jehož výstup zapíše do uvedeného souboru)<br>`$ at -mf seznam 20:00 25.06.2007`<br>(provede v daný čas seznam příkazů uvedených v souboru a po dokončení úlohy odešle uživateli email) |
| **atq** \| **at -l** | vypíše seznam naplánovaných úloh (číslo úlohy, čas spuštění, jméno uživatele, jenž úlohu naplánoval) |
| **atrm** \| **at -d** <cislo_ulohy ...> | odstraní naplánovanou úlohu |
| **lpr** [<soubor>] | tiskne soubor; není-li soubor uveden, čte ze STDIN, **-P** <tiskarna> určí tiskárnu, **-#** <n> určí počet kopií |
| **lpq** | vypíše úlohy v tiskové frontě výchozí tiskárny, **-P** <tiskarna> určí tiskárnu, **-a** úlohy na všech tiskárnách, **-l** podrobný výpis |
| **lprm** [<cislo_ulohy> \| **-**] | odstraní danou úlohu z tiskové fronty či celou frontu, **-P** <tiskarna> určí tiskárnu; bez argumentu odstraní současnou úlohu na výchozí tiskárně |

| **ÚLOHY** | |
|---|---|
| **crontab** [<soubor>] | nastaví existující soubor pro pravidelné spouštění uvedených úloh prostřednictvím démona cron, **-e** vytvoří či edituje crontabový soubor, **-l** vypíše obsah crontabového souboru, **-r** odstraní crontabový soubor, **-u** <uzivatel> nastaví crontabový soubor daného uživatele; v souboru se úlohy uvádí každá na zvláštní řádek, který se u uživatelských souborů skládá z 6 polí oddělených mezerami – 5 pro zadání času (v pořadí minuta, hodina, den v měsíci, měsíc a den v týdnu (ne–so = 0–6), „*“ vyhovuje všem hodnotám, „*/“<n> každý *n*tý časový interval, „,“ odděluje hodnoty téhož pole, „-“ označuje rozsah hodnot) a 1 pro příkaz samotný (pro zadání příkazu se používá absolutní cesta, nejde-li o vestavěný příkaz nebo klíčové slovo shellu, má-li nějaký výstup, odešle se uživateli emailem); prázdný řádek a řádek začínající znakem „#“ se ignoruje; mezi systémové soubory patří */etc/crontab* a soubory umístěné v */etc/cron.d*, které obsahují 1 pole navíc mezi časovým zadáním a příkazem – uživatele, pod kterým se má daná úloha spustit; soubory uživatelů se nachází ve */var/spool/cron/<uzivatel>*, a pokud jsou editovány ručně (nikoliv příkazem „crontab -e“), nejsou změny automaticky démonem načteny; oprávnění uživatelé jsou uvedeni v */etc/cron.allow*, neoprávnění uživatelé v */etc/cron.deny*, existují-li oba soubory, */etc/cron.deny* se ignoruje<br>*30 8 * * * /usr/bin/df \| /bin/mail admin*<br>(spustí úlohu každý den v 8:30)<br>**/30 8-16 * * 1-5 /usr/bin/who >> /tmp/users.out*<br>(spustí úlohu každých 30 min v pracovní době)<br>*0 22 * * 5 /usr/local/scripts/backup.sh*<br>(spustí skript každý pátek ve 22:00)<br>*0 0 1 * * for file in $(/bin/find /web/logs/*); do > $file; done*<br>(spustí úlohu na začátku každého měsíce) |

From:
https://www.prompt.cz/ - **Prompt.cz**

Permanent link:
**https://www.prompt.cz/procesy-a-ulohy**

Last update: **2025/07/18 12:49**