**۵ راه برای حفظ SSH Sessions از راه دور و اجرای پردازش ها پس از قطع اتصال**

SSH یا Secure Shell در اصطلاحات ساده راهی است که فرد می تواند از راه دور به یک کاربر دیگر در سیستم دیگری دسترسی پیدا کند، اما تنها در خط فرمان، یعنی حالت غیر GUI. در شرایط فنی تر، زمانی که ما اتصال SSH بر روی یک کاربر در برخی از سیستم ها و اجرای دستورات در آن دستگاه انجام می دهیم،در واقع یک شبه ترمینال ایجاد می کنیم و آن را به پوسته ورود منتقل می کنیم.

هنگامی که ما session را می بندیم یا زمان session تمام می شوید و یا session بر قرار است و کاری تنجام نمی دهیم، سیگنال SIGHUP به شبه ترمینال ارسال می شود و تمام jobs هایی که در آن ترمینال اجرا می شوند، حتی jobs هایی که شاخه های اصلی را دارند در حالیکه در شبه ترمینال آغاز شده نیز سیگنال SIGHUP ارسال می شود و مجبور به خاتمه دادن دستور می شوند.

فقط jobs هایی که پیکربندی شده اند تا این سیگنال را نادیده بگیرند در terminal باقی می مانند. در سیستم های لینوکس، راه های زیادی برای اجرای این Job ها در هنگام اتصال از راه دور به سرور و یا هر ماشین دیگری حتی بعد از قطع شدن session هست.

درک فرآیندهای در لینوکس فرآیند طبیعی (Normal Process)فرایندهایی طبیعی هستند که طول عمر یک session را دارند. آنها در طول session به عنوان فرایندهای پیش زمینه شروع میشوند و در زمان خاصی به پایان میرسند یا زمانی که session بسته می شود.صاحب این فرآیندها به عنوان یکی از کاربر ها معتبر سیستم مانند root می باشند.

فرآیند یتیم (Orphan Process)

فرایندهایی یتیم هستند که در ابتدا به وسیله یک پردازش والد (parent) اجرا شده اند و به هر دلیلی آن فرآیند والد از بین رفته است و این فرآیند یا پردازش باقی می ماند. چنین فرایندهایی، init را به عنوان والد فوری خود در نظر می گیرند و تا زمانی که از بین بروند و یا تمام شوند در این فرآیند می مانند.

Daemon Process

این ها برخی از فرآیندهای به طور عمدی یتیم هستند؛ فرآیندهایی هستند که به طور عمدی باقی می مانند در حال اجرا هستند و به عنوان daemon شناخته می شوند و یا به طور عمدی محصولات فرعی یتیم شده اند. آنها معمولا فرآیندهای طولانی مدت هستند که یک بار شروع شده و سپس از هر ترمینال کنترل جدا می شوند تا بتوانند در پس زمینه اجرا شوند تا اینکه آنها کامل شوند و یا نتوانند خطا را انجام دهند. والدین این فرایندها عمدا میمیرند و فرزندان را در پس زمینه اجرا می کنند.

تکنیک هایی برای نگه داشتن SSH Session پس از قطع اتصال

۱. استفاده از فرمان Screen برای نگه داشتن جلسات SSH در حال اجرا

screen یک Window Manager متنی برای لینوکس است که به کاربر اجازه می دهد تا همزمان مدیریت sessions ترمینال چندگانه، تعویض بین sessions، ثبت sessions برای جلسات در حال اجرا روی صفحه، و حتی بازگرداندن session در هر زمانی که ما بدون هیچ گونه نگرانی در مورد جلسه ورودی خارج یا ترمینال بسته می شود.

از طریق session های screen می توانید فرآیندی را اجرا کنید و آن را رها کنید تا در background اجرا شود و در هر زمان و مکانی که می خواهد دوباره به آن دسترسی داشته باشید. فقط کافیست session خود در screen را اجرا کنید و در هر وقت که می خواهید از سیستم خارج شوید و یا کنترل terminal را قطع کنید و هنگامی که نیاز به دسترسی دوباره داشتید می توانید session خود را ادامه دهید.

شروع یک session با screen

|  |  |
| --- | --- |
|  |  |



هنگامی که session روی screen آغاز شد، میتوانید هر دستور را اجرا کنید و session را با جدا کردن جلسه ادامه دهید.



قطع کردن یک Screen

فقط زمانی که می خواهید از جلسه از راه دور خارج شوید، اما می خواهید جلسه ای که در آن دستگاه ایجاد کرده اید، حفظ شود،تنها کاری که باید انجام دهید این است که Screen را از ترمینال جدا کنید تا هیچ ترمینال کنترل وجود نداشته باشد. پس از انجام این کار، می توانید با خیال راحت از سیستم خارج شوید.

برای جدا شدن یک Screen از ترمینال از راه دور، فقط “Ctrl + a” را بلافاصله پس از “d” فشار دهید و پیامی را در صفحه جدا شده است، مشاهده می کنید. اکنون میتوانید بیرون بیایید و session شما زنده بماند.



ادامه دادن Session در Screen

اگر میخواهید به یک Session قطع شده که قبلا از آن خارج شده اید، مجددا وارد شوید و فقط یک صفحه باز شود، “screen -r” تایپ کنید و اگر می خواهد چندین Session را باز کنید از دستور “screen -r <pid.tty.host>” استفاده کنید.

|  |  |
| --- | --- |
| 12 | $ screen -r$ screen -r &lt;pid.tty.host&gt; |



۲. استفاده از Tmux (Terminal Multiplexer)  برای حفظ SSH Sessions در حال اجرا

Tmux یک نرم افزار دیگر است که جایگزینی برای screen است. ویژگی های بسیاری از screen را دارد و دارای قابلیت های اضافی دیگری است که از screen قدرتمند تر است.

از دیگر امکانات این نرم افزار تقسیم صفحه به صورت افقی و عمودی بین چند پنجره ، تغییر اندازه پنجره ، کنترل برقراری ارتباط session ، استفاده از Script در خط فرمان و … .با توجه به این ویژگی های tmux، تقریبا  در همه ی توزیع های یونیکس استفاده شده است و حتی در سیستم پایه OpenBSD نیز موجود است.

شروع کار با Tmux

|  |  |
| --- | --- |
| 1 | $ tmux |



پس از انجام عملیات خود در ترمینال، می توانید این session را از ترمینال کنترل جدا کنید تا آن را در پس زمینه اجرا کنید و شما می توانید با خیال راحت از سیستم خارج شوید.



جدا سازی Tmux Session از Terminal

شما می توانید “tmux detach” را در جلسه tmux اجرا کنید یا می توانید از میانبر (Ctrl + b سپس d) استفاده کنید. بعد از این، session فعلی شما جدا می شود و شما به ترمینال خود بازگردید و از آنجا می توانید با امنیت بیرون بیایید.

|  |  |
| --- | --- |
| 1 | $ tmux detach |



۳.استفاده از nohup برای حفظ SSH Sessions در حال اجرا

اگر شما با screen یا tmux آشنا نیستید، می توانید از nohup استفاده کنید و دستور فرمان طولانی خود را به پس زمینه ارسال کنید تا بتوانید به کار خود ادامه دهید در حالی که فرمان در حال اجرا در پس زمینه نگه داشته می شود. پس از آن شما می توانید با خیال راحت از سیستم خارج شوید.

با فرمان nohup، ما فرایند را برای نادیده گرفتن سیگنال SIGHUP که از طریق session ssh در خاتمه ارسال می کنیم نادیده می گیریم، بنابراین فرمان حتی پس از خروج از session نیز ادامه می یابد. در خروج از session فرمان از کنترل ترمینال جدا شده و در حال اجرا در پس زمینه به عنوان پردازش daemon نگه می دارد.

اجرای فرمان در پس زمینه با استفاده از nohup

در اینجا، یک سناریوی ساده است که در آن، ما دستور find را برای جستجوی فایل ها در پس زمینه در session ssh با استفاده از nohup اجرا می کنیم، پس از آن task به پس زمینه فرستاده می شود و بلافاصله پس از ارسال PID و شناسه شغلی از پردازش ([JOBID] PID ) lahin مشاهده می کنید.

|  |  |
| --- | --- |
| 1 | # nohup find / -type f $gt; files\_in\_system.out 2&gt;1 &amp; |



بازنگری session برای مشاهده اگر job هنوز در حال اجرا است

هنگامی که شما مجددا وارد سیستم شوید، می توانید وضعیت دستور را بررسی کنید،با استفاده از fg٪ JOBID به پیش زمینه بازگردانید تا مانع پیشرفت و … شود. در پایین، خروجی نشان می دهد که job کامل شده و در ورود مجدد نشان داده نشده و خروجی که نمایش داده شده است.

|  |  |
| --- | --- |
| 1 | # fg %JOBID |



**۴.استفاده از disown برای حفظ SSH Sessions در حال اجرا**

یکی دیگر از روش ها برای اجازه دادن به فرمان شما یا اجرا یک task در پس زمینه و حتی پس از خروج از session یا قطع ارتباط با استفاده از دستور disown است.

Disown، برای شما job را از لیست پردازش ها حذف می کند، بنابراین فرایند هنگام قطع شدن session محافظت می شود زیرا SIGHUP توسط پوسته در هنگام خروج از سیستم شما دریافت نمی شود.

معایب این روش اینست که باید آن را فقط برای job هایی که نیازی به هیچ ورودی از stdin ندارند و نیازی به نوشتن در stdout ندارند استفاده کنید، مگر آنکه به طور خاص تغییرات ورود و خروجی job را هدایت کنید، زیرا زمانی که job سعی در برقراری ارتباط با stdin یا stdout، آن را متوقف خواهد کرد.

اجرای دستور با استفاده از disown در پس زمینهدر زیر، ما دستور ping را به پس زمینه فرستادیم تا ut در حال اجرا باشد و از لیست کار حذف شود. همانطور که مشاهده شد، این کار برای اولین بار به حالت تعلیق درآمده بود، پس از آن هنوز در لیست کار به عنوان شناسه فرآیند: 15368 بود.

|  |  |
| --- | --- |
| 1234 | $ ping tecmint.com &gt; pingout &amp;$ jobs -l$ diswon -h %1$ ps -ef | grep ping |



پس از disown سیگنال عزل به job منتقل شد و از لیست کار حذف شد، هر چند هنوز در پس زمینه اجرا می شد. این کار زمانی انجام می شود که مجددا به سرور راه دور وارد شوید، همانطور که در زیر مشاهده می کنید.

|  |  |
| --- | --- |
| 1 | $ ps -ef | grep ping |



۵.استفاده از setsid برای حفظ SSH Sessions در حال اجرا

یکی دیگر از ابزارهای مورد استفاده setsid است. Nohup دارای معایبی است به این معنی که گروه فرایند باقی می ماند همانطور که روند در حال اجرا با nohup برای هر سیگنال فرستاده شده به کل گروه فرآیند (مانند Ctrl + C) آسیب پذیر است.
از سوی دیگر،setsid یک گروه فرآیند جدید را به فرآیند اضافه می کند و از این رو، فرایند ایجاد شده کاملا در یک گروه فرایند جدید قرار می گیرد و می تواند بدون ترس از از بین رفتن حتی بعد از خروج از session، اجرا شود.

اجرا هر دستور را با استفاده از setid

در اینجا، نشان می دهد که روند ‘sleep 10m’ از زمان ترمینال کنترل جدا شده است.

|  |  |
| --- | --- |
| 12 | $ setsid sleep 10m$ ps -ef | grep sleep |



در حال حاضر، زمانی که مجددا وارد جلسه می شوید، هنوز این روند را در حال اجرا خواهد بود.

|  |  |
| --- | --- |
| 1 | $ ps -ef | grep [s]leep |

