

Real time signals

לפני שנסביר על סוג סיגנלים זה נסביר בכלליות מהו אות.

אות הינו הפרעה בתוכנה שנשלחת ע"י הגרעין לתהליך כדי לדווח על מצב חריג או למשל לדווח על ניסיון של תהליך לגשת למיקום לא חוקי, או לכל אירוע אסינכרוני אחר.

אותות הינן אמצעי לתקשר בין תהליכים (ישנם עוד כמה כגון צינורות סוקטים וכו'), כשתהליך מקבל אות הוא עוצר את הפעולה שלו ומגיב לאות שקיבל בהתאם למה שמוגדר האות(למשל

SIGKILL אות המשמש להרוג אות וכו')

ישנם כמה חסרונות לשימוש באותות רגילים והמשמעותי שבהם הוא שזה מתנהג בצורה שונה בגרסאות שונות של לינוקס\ יונוקס.

התנהגות לא מוגדרת אם ה signal handler כבר פועל- האותות לא חסומים בזמן ביצוע פעולה שלו.

אין אפשרות להעביר משתנים ל handler.

ז.א שאם נשלח אות למערכת בזמן שהוא מטפל באות אחר יש סיכוי שהאות לא יתקבל ויטופל כלל.

לעומת זאת לאותות בזמן אמת - אין שימוש ייעודי המוגדר מראש אנו משתמשים בהם למטרות שלנו , המצב הדיפולטיבי הינו להפסיק תהליך קבלה.

היתרונות בכך הוא שניתן להכניס לתור כמה אותות במקביל אם כבר יש אות בטיפול , האותות ייכנסו לתור וישמרו בניגוד לאותות רגילים שיכולים להיעלם.

יתרון נוסף וחשוב הינו כי אותות בממן אמת מועברים בסדר שנשלחו , לעומת אותות רגילים שאם ממתנים כמה אותות לתהליך סדר ההעברה שלהם אינו מוגדר.

החסרונות של אותות בזמן אמת הם:

עדיפות - לינוקס ויישומים אחרים נותנים עדיפות לאותות רגילים על פני אותות בזמן אמת.(יתרון לאות רגיל , מכיוון שהוא עלול להיאבד ולא נשמר בתור כמו אותות בזמן אמת המערכת תתן לט עדיפות).

קושי - יצירת אותות בזמן אמת זה דבר מסובך המכיל שליטה בהמון דברים שצריך לדעת ולכן שימוש לא נכון עלול לגרום להמון בעיות.

כעת נסביר על 2 סוגים של אות בזמן אמת :

1) סיגנל זה הינו סיגנל מיוחד הנותן לנו אפשרות לקבוע מה בדיוק יבצע הסיגנל שלנו-Sigaction

הוא מספק מגוון עצום של אפשרויות תוך כדי טיפול באותות, אך עם מורכבות.

המבנה שלו הוא `int sigaction(int signum, const struct sigaction *act, struct sigaction *oldact)` הארגומנט הראשון הינו האות שבו אנו מטפלים, השלישי הינו הגדרת ההתנהגות הקודמת של האות , והשני הוא מצביע לחתימה שהוא מבנה של נתוני גרעין- ז.א. הפעולה החדשה שאנו רוצים שהאות יבצע.

באמצעות כך ניתן לטפל באותו באופן מדויק ובטוח יותר יחד עם ניידות ושליטה בחריגות או לחורי לולאה שנוצרו תוך שימוש בשיחת מערכת.

ניתן להשתמש בו גם כדי לבדוק אם אות נתון תקף למחשב הנוכחי על ידי שליחת נאל. בארגומנטים השני והשלישי.

חסרונות :

כששולחים אותות עם `SA_SIGINFO` handler

אזי הגרעין לא תמיד יחזיר ערכים רלוונטיים לאות זה.
בנוסף הבנייה של אות זה הינה מורכבת ולכן יכול ליצור בעיות במימוש וכדומה.

אות זה משמש להכנסת אות לתור של תהליך. -Sigqueue

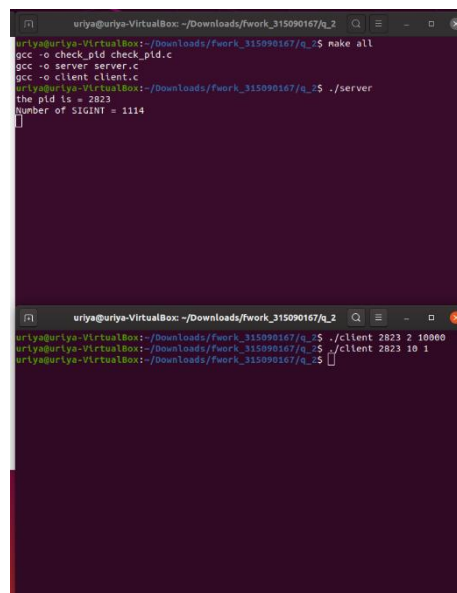
שולחים אות עם פרמטר שהוא מצויין לתהליך מסוים – אם המשאבים זמינים בתהליך אזי הוא יקבל את האות ויתחיל לפעול , אם לתהליך יש מעבד אותות אז הוא יכול לקבל את המידע שנשלח באמצעות handler. Signal

המבנה שלו(int sigqueue(pid_t pid s, int signo ,const union sigval value)

כאשר הארגומנט הראשון הינו המספר המזהה של התהליך(או קבוצה של תהליכים) , הארגומנט השני הוא סוג האות שנשלח , הארגומנט השלישי הוא הערך שמועבר על ידי המשתמש עם האות לקבלה על ידי התהליך.

דוגמא להרצת הקוד בשאלה :

כפי שניתן לראות שלחתי 10000 אותות אולם התקבלו רק 1114.



```
uriya@uriya-VirtualBox: ~/Downloads/fwork_315090167/q_2
uriya@uriya-VirtualBox:~/Downloads/fwork_315090167/q_2$ make all
gcc -o check_pid check_pid.c
gcc -o server server.c
gcc -o client client.c
uriya@uriya-VirtualBox:~/Downloads/fwork_315090167/q_2$ ./server
the pid is = 2823
number of signal = 1114

uriya@uriya-VirtualBox:~/Downloads/fwork_315090167/q_2$
uriya@uriya-VirtualBox:~/Downloads/fwork_315090167/q_2$ ./client 2823 2 10000
uriya@uriya-VirtualBox:~/Downloads/fwork_315090167/q_2$
```