

כיצד פועל השתרשות: הסבר טכני של תהליך ההשתרשות באנדרואיד

תמיד הייתי סקרן איך ההשתרשות באמת עובדת Eee Pad Slider מאחורי הקלעים. לאחר שרכשתי לאחרונה חדש, טאבלט חלת דבש שעד כה אף לא הצליח לבצע רוט, התסכול בלהיות נעול מחוץ לחלק החומרה המדהים הזה עם כל כך הרבה פוטנציאל הוביל אותי סוף סוף לשבת ולהבין מה בדיוק פירוש השתרשות, מה זה כרוך מנקודת מבט טכנית, וכיצד האקרים בטבע מתקרבים להשרשה של מכשיר חדש. למרות שכל המידע הזה נמצא בחוץ, לא הצלחתי למצוא מאמר טוב שיש בו גם את רמת הפירוט הטכני שרציתי וגם מבוא מתאים לתמונה הגדולה, ולכן החלטתי לכתוב משלי.

לשורש מכשיר noob-זה לא מדריך ידידותי ל אנדרואיד מסוים. במקום זאת, זהו הסבר כללי על מנסים למנוע ROM האופן שבו רכיבי אנדרואיד גישה בלתי-פריבילגית, כיצד האקרים תוקפים את הבעיה הזו וכיצד תוכנות השתרשות ממנפות ניצולים שונים כדי להביס את מנגנוני האבטחה הללו.

1. המטרה

צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות

צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות



תחילה ניקח צעד אחורה ונחשוב בדיוק למה אנו ROMs מתכוונים בהשרשה. תשכחו מהבהב של WiFi מותאמים אישית, הפעלת שיתוף אינטרנט בין ביסודו, השתרשות Superuser.apk; או התקנת עוסקת בהשגת גישת שורש למערכת הלינוקס הבסיסית מתחת לאנדרואיד ובכך להשיג שליטה מוחלטת על התוכנה שפועלת במכשיר. דברים שדורשים גישת שורש במערכת לינוקס טיפוסית - הרכבה והסרה של מערכות קבצים, הפעלת שרתי proxy או DNS או DHCP או HTTP או SSH chroot-ing, המועדפים עליך, הרג תהליכי מערכת וכו' - דורשים גישת שורש גם באנדרואיד. היכולת להריץ פקודות שרירותיות כמשתמש השורש מאפשרת לך לעשות הכל על מערכת לינוקס / אנדרואיד, וזוהי המטרה האמיתית של השתרשות

Stock OEM Android builds typically do not allow users to execute arbitrary code as root. This essentially means that you as a user are granted only limited control over your own device; you can make your device do task X only if the manufacturer explicitly decided to allow it and shipped a program to do it. You will not be able to use third-party apps to accomplish a task that your manufacturer does not wish you to do. WiFi tethering is a good example of this. Cell phone carriers obviously do not want you to tether your phone without paying them additional charges. Therefore, many phones come pre-packaged with their own proprietary WiFi tethering apps that demand extraneous fees. But without root access, you will not be able to install a free alternative like [Wireless Tether For Root Users](#). Why this is accepted practice in the industry is a mystery to me. The only difference between cell phones,

צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות



tablets and computers is their form factor; but while a PC vendor would fail spectacularly if they tried to prevent users from running arbitrary programs on their machines, cell phone vendors are clearly not judged along the same lines. But such arguments would belong to another article.

II. The Enemy: Protection Mechanisms On A Stock OEM Android ROM

Bootloader and Recovery

The bootloader, the first piece of code executed when your device is powered on, is responsible for loading the Android OS and the recovery system and flashing a new ROM. People refer to some bootloaders as "unlocked" if a user can flash and boot arbitrary ROMs without hacking; unfortunately, many Android devices have locked bootloaders that you would have to hack around in order to make them do anything other than boot the stock ROM. A Samsung smartphone I had used some months ago had an unlocked bootloader; I could press a certain combination of hardware keys on the phone, connect it to my computer, and flash any custom ROM onto it using Samsung's utilities without having to circumvent any protection mechanisms. The same is not true for my Motorola Droid 2 Global; the bootloader, as far as I know, cannot be hacked. The Eee

צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות

Pad Slider, on the other hand, is an interesting beast; as with other nVidia Tegra 2 based devices, its bootloader is controllable through the `nvflash` utility, but only if you know the *secure boot key* (SBK) of the device. (The SBK is a private AES key used to encrypt the commands sent to the bootloader; the bootloader will only accept the command if it has been encrypted by the particular key of the device.) Currently, as the SBK of the Eee Pad Slider is not publicly known, the bootloader remains inaccessible.

System recovery is the second piece of low-level code on board any Android device. It is separate from the Android userland and is typically located on its own partition; it is usually booted by the bootloader when you press a certain combination of hardware keys. It is important to understand that it is a totally independent program; Linux and the Android userland is not loaded when you boot into recovery, and any high-level concept such as root does not exist here. It is simple program that really is a very primitive OS, and it has absolute control over the system and will do anything you want as long as the code to do it is built in. Stock recovery varies with the manufacturer, but often includes functionalities like reformatting the `/data` partition (factory reset) and flashing an update ROM (`update.zip`, located at the root of the external microSD card) signed by the manufacturer. Note I said *signed by the manufacturer*; typically it is not possible to



צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות



flash custom update files unless you obtain the private key of the manufacturer and sign your custom update with it, which is both impossible for most and illegal under certain jurisdictions. However, since recovery is stored in a partition just like `/system`, `/data` and `/cache` (more about that later), you can replace it with a custom recovery if you have root access in Linux / Android. Most people do just that upon rooting their device; [ClockworkMod Recovery](#) is a popular third-party recovery image, and allows you to flash arbitrary ROMs, backup and restore partitions, and lots of other magic.

ADB

ADB (see [the official documentation for ADB](#)) allows a PC or a Mac to connect to an Android device and perform certain operations. One such operation is to launch a simple shell on the device, using the command `adb shell`. The real question is what user do the commands executed by that shell process run as. It turns out that it depends on the value of an Android system property, named `ro.secure`. (You can view the value of this property by typing `getprop ro.secure` either through an ADB shell or on a terminal emulator on the device.) If `ro.secure=0`, an ADB shell will run commands as the root user on the device. But if `ro.secure=1`, an ADB shell will run commands as an unprivileged user on the device. Guess what `ro.secure` is set to on almost every stock OEM Android build. But

צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות



can we change the value of `ro.secure` on a system? The answer is no, as implied by the `ro` in the name of the property. The value of this property is set at boot time from the `default.prop` file in the root directory. The contents of the root directory are essentially *copied* from a partition in the internal storage on boot, but you cannot write to the partition if you are not already root. In other words, this property denies root access via ADB, and the only way you could change it is by gaining root access in the first place. Thus, it is secure.

Android UI

On an Android system, all Android applications that you can see or interact with directly are running as `_un_privileged` users in sandboxes. Logically, a program running as an unprivileged user cannot start another program that is run as the privileged user; otherwise any program can simply start another copy of itself in privileged mode and gain privileged access to everything. On the other hand, a program running as root can start another program as root or as an unprivileged user. On Linux, privilege escalation is usually accomplished via the `su` and `sudo` programs; they are often the only programs in the system that are able to execute the system call `setuid(0)` that changes the current program from running as an unprivileged user to running as root. Apps that label themselves as requiring root are in reality just executing other programs

צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות



(often just native binaries packaged with the app) through `su`. Unsurprisingly, stock OEM ROMs never come with these `su`. You cannot just download it or copy it over either; it needs to have its SUID bit set, which indicates to the system that the programs this allowed to escalate its runtime privileges to root. But of course, if you are not root, you cannot set the SUID bit on a program. To summarize, what this means is that any program that you can interact with on Android (and hence running in unprivileged mode) is unable to either 1) gain privileged access and execute in privileged mode, or 2) start another program that executes in privileged mode. If this holds, the Android system by itself is pretty much immune to privilege escalation attempts. We will see the loophole exploited by on-device rooting applications in the next section.

III. Fighting the System

So how the hell do you root an Android? Well, from the security mechanisms described above, we can figure out how to attack each component in turn.

If your device happens to have an unlocked bootloader, you're pretty much done. An example is the Samsung phone that I had had. Since the bootloader allowed the flashing of arbitrary ROMs, somebody essentially pulled the stock ROM from the phone (using `dd`), added `su`, and repackaged it into a modified ROM. All I as a

צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות

user needed to do was to power off the phone, press a certain combination of hardware keys to start the phone in flashing mode, and use Samsung's utilities to flash the modified ROM onto the phone.

Believe it or not, certain manufacturers don't actually set `ro.secure` to 1. If that is the case, rooting is even easier; just plug the phone into your computer and run ADB, and you now have a shell that can execute any program as root. You can then mount `/system` as read-write, install `su` and all your dreams have come true.

אבל מכשירי אנדרואיד רבים אחרים נועלים אתחול כפי שהוסבר לעיל, הם לא `ro.secure` ונעילה אמורים להיות בעלי יכולת שורש מכיוון שאתה יכול לקיים אינטראקציה רק עם תוכניות חסרות פריבילגיה במערכת והם לא יכולים לעזור לך לבצע קוד מיוחד כלשהו. אז מה הפתרון?

אנו יודעים שמספר תוכניות חשובות, כולל שירותי מערכת ברמה נמוכה, חייבות לפעול כשורש אפילו באנדרואיד כדי לגשת למשאבי חומרה. הקלדה או אמולטור ADB או דרך) על מעטפת אנדרואיד `ps` תיתן לכם מושג. תוכניות אלה (מסופך במכשיר התהליך הראשון, `init` מתחילות על ידי התהליך לעתים קרובות אני מרגיש) שהתחיל על ידי הגרעין הם די דומים לאדם וחווה - `init` שהגרעין והתהליך ואז `init`, בצורה מסוימת `init` הגרעין משריץ שצריכים (ממשיך ומוליד את כל השאר תהליכים לפעול כשורש מכיוון שהוא צריך להתחיל תהליכי מערכת מיוחדים אחרים.

כעת הנה התובנה העיקרית: אם אתה יכול לפרוץ / להערים על אחד מתהליכי המערכת הללו הפועלים במצב מיוחד כדי לבצע את הקוד השרירותי שלך,



צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות

זה עתה השגת גישה מיוחדת למערכת. כך פועלות z4root, כל שיטות השורש בלחיצה אחת, כולל וכו'. אם אתה באמת סקרן, אני gingerbreak ממליץ בחום על **המצגת המצוינת הזו על המנצלים השונים המשמשים את כלי ההשרשה הנוכחיים**, אבל הפרטים אינם רלוונטיים כאן כמו הרעיון הפשוט שמאחוריהם. הרעיון הזה הוא שישנן root-נקודות תורפה בתהליכי המערכת הפועלים כ ברקע, שאם ניצלו, יאפשרו לנו להפעיל קוד ובכן, אותו "קוד שרירותי" הוא ללא root-שרירותי כ במצב `/system` ספק קטע קוד שנטען באופן קבוע `su` קריאה-כתיבה ומתקין עותק של במערכת, כך שמאז ואילך לא נצטרך לקפוץ בין החישוקים כדי להפעיל את התוכניות מאוד רצינו לרוץ מלכתחילה.

מכיוון שאנדרואיד הוא קוד פתוח כמו לינוקס, מה שאנשים עשו זה לבחון ולנמק את קוד המקור של שירותי המערכת השונים עד שהם מוצאים חור אבטחה שהם יכולים למנף. זה הופך להיות קשה יותר ככל שגוגל והמתחזקים של פיסות הקוד השונות מתקנים את הפגיעויות הספציפיות הללו כשהן מתגלות ומתפרסמות, מה שאומר שהניצול בסופו של דבר יתיישן עם מכשירים חדשים יותר. אבל החדשות הטובות הן שהיצרנים לא מספיק כדי לתקן פגיעות OTA טיפשים כדי לדחוף עדכוני רק כדי למנוע השתרשות מכיוון שזה מאוד יקר עבורם; בנוסף, המכשירים בשוק תמיד מפגרים אחרי מהדורות התוכנה החדשות ביותר. לפיכך, לוקח לא מעט זמן עד שכלי ההשרשה הללו הופכים חסרי תועלת על ידי תיקונים חדשים, ועד אז יש לקוות שמנצלים אחרים היו מתגלים.

IV. ראה את זה בפעולה!



כדי לראות את כל זה בפעולה, אתה מוזמן לעיין במאמר ההמשך שלי: [השתרשות אנדרואיד: מדריך למפתחים](#), המסביר כיצד יישמתי את החומר הזה כדי להבין כיצד לבצע רוט למכשיר בפועל.

לר: אין טר בוק וייל

ALSO ON JICHU4N.COM

צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות

10 years ago · 13 comments
How X Window Managers Work, And ...

10 years ago · 6 co
DEBUG trağ PROMPT

48 Comments

Login

Join the discussion...

LOG IN WITH



OR SIGN UP WITH DISQUS ?

Name

11 [Share](#)

[Best](#) [Newest](#) [Oldest](#)

N nisar



צ'ואן ג'י



על אודות

פרויקטים

מאמרים

הערות טכניות