Western Digital.

WHITE PAPER JUNE 2020

Security Unlock Command at Linux Kernel Level

Author | Wes McMillen, Western Digital Corporation

Contents

Overview

Access to user data on an ATA HDD can be restricted by enabling security via the ATA Security Feature Set. Once security has been enabled and a password has been set, the HDD will be locked upon power-on. All access to user data will be restricted on a locked HDD until the HDD has been unlocked. A HDD can be unlocked by sending a SECURITY UNLOCK command with the correct password to the HDD.

Some systems running Linux do not have built-in support for unlocking HDDs preventing this HDD security feature from being utilized.

This document will provide an example of how to send a SECURITY UNLOCK command to the HDD at the Linux Kernel Level so locked HDDs can be utilized in Linux. It assumes working knowledge of the ATA Security Feature Set and does not attempt to explain it or how to lock a HDD.

Implementation

Issuing Security Unlock at Kernel Level

While unlocking the HDD can be implemented at the Application or Kernel level, it is recommended to be implemented at the Kernel Level because implementing it at the Kernel Level closes a window where commands could be sent to the HDD prior to the HDD being unlocked. Those commands sent to the HDD prior to the unlock may return an error and cause unwanted system side effects (for example, the system may switch from using NCQ commands to Non-NCQ commands because NCQ command(s) were aborted prior to the HDD being unlocked).

Password Generation

A password must be supplied when a SECURITY UNLOCK command is sent to the HDD. Because the password should be unique per system (for security reasons), Western Digital cannot provide this code. The user must provide the code to generate the password prior to the SECURITY UNLOCK command being sent. The location to insert the password generation code is referenced in the sample code provided below.

Sample Linux Kernel Modification

Step 1:

Find the `libata-core.c` file in the linux kernel source tree (usually under [source root]\drivers\ata).

Note, all edits will be made to this file.

Step 2:

Copy following block of code to top section of `libata-core.c` (below headers includes).

```
bool ata_id_security_enabled(const u16 *id);
bool ata_id_security_locked(const u16 *id);

enum {
    USER_PWD_LEN = 32
};

bool ata_id_security_enabled(const u16 *id)
{
    /* Make sure Security Feature Set is supported */
    if ((id[ATA_ID_COMMAND_SET_1] & 0x0002) != 0x0002)
        return false;

    /* Return Security Enabled State */
    return id[ATA_ID_CFS_ENABLE_1] & (1 << 1);
}

bool ata_id_security_locked(const u16 *id)
{
    /* Make sure Security Feature Set is supported */
    if ((id[ATA_ID_COMMAND_SET_1] & 0x0002) != 0x0002)
        return false;

    /* Return Security Lock State */
    return id[ATA_ID_DLF] & (1 << 2);
}

/**
 *  ata_security_unlock_with_user_pwd
 *  @dev: target device
 *  @user_pwd: password string
 *
 *  Attempts to unlock @dev with user password @user_pwd
 *
 *  LOCKING:
 *  Kernel thread context (may sleep)
 *
 *  RETURNS:
 *  0 on success, -errno otherwise.
 */
```

```
unsigned int ata _ security _ unlock(struct ata _ device *dev)
{
    unsigned int   rc = 0;
    unsigned char *pwd _ buf = NULL;

    if (dev && dev->id) {
        /* check whether HDD is locked */
        if (ata _ id _ security _ enabled(dev->id) && ata _ id _ security _
locked(dev->id))
{
            pwd _ buf = kzalloc(USER _ PWD _ LEN + 1, GFP _ NOIO);

            if (!pwd _ buf) {
                rc = -ENOMEM;
            }
            else {

   /* ***************** IMPORTANT - Read This ******************
        * Code must be inserted here to populate the pwd _ buf with the
        * correct HDD password prior to the proceeding call.
      ************************************************************
    */

         rc = ata _ security _ unlock _ with _ user _ pwd(dev, pwd _ buf);

                kfree(pwd _ buf);
            }
        }
    }
    else {
        rc = -ENODEV;
    }

    ata _ dev _ dbg(dev, "ata _ security _ unlock(), rc=%x\n", rc);

    return rc;
}
```

Sample Linux Kernel Modification (Cont.)

Step 3:

In the code block that was copied above in Step 2, find the commented section **IMPORTANT - Read This**. Note, the sample code provided by Western Digital does not include any code to generate a password. The user must implement this code. Add code directly beneath the commented section that populates the HDD password in the **pwd _ buf** buffer.

Step 4:

In **libata-core.c**, within the function int ata_dev_configure(struct ata_device *dev), find the snippet of code below:

```
    ...

    /* let ACPI work its magic */
    rc = ata _ acpi _ on _ devcfg(dev);
    if (rc)
        return rc;

    ...
```

Copy the following code directly underneath the code found above:

```
   /* === unlock HDD if needed */
    int unlockStatus = ata _ security _ unlock(dev);
    if (unlockStatus) {
        ata _ dev _ dbg(dev, “ata _ security _ unlockI) failed, unlockStatus =
%x\n”, unlockStatus);
    }
```

The complete section should look like this:

```
    ...

    /* let ACPI work its magic */
    rc = ata _ acpi _ on _ devcfg(dev);
    if (rc)
        return rc;

    /* === unlock HDD if needed */
    int unlockStatus = ata _ security _ unlock(dev);
    if (unlockStatus) {
        ata _ dev _ dbg(dev, “ata _ security _ unlockI) failed, unlockStatus =
%x\n”, unlockStatus);
    }
    ...
```