Watchdog Bypass Techniques

Douglas McKee
@fulmetalpackets

The McAfee Advanced Threat Research team conducts security research with the aim of staying ahead of the evolving threat landscape to expose and reduce attack surfaces. This series of white papers discusses laboratory security research techniques that are generally known among the professional community of security researchers. The white papers are provided to elevate collaboration and security within the industry and are not to be used for unlawful purposes. Security researchers are responsible for lawfully obtaining equipment and for complying with contracts and licenses for their research.

When dealing with embedded systems it is not uncommon to run across a software and/or hardware watchdog timer. For researchers trying to dynamically reverse engineer a platform this can quickly become a problem. So how do we overcome this? Unfortunately, there is no one method that will always work, but there are several methods that may allow you to disable or bypass a watchdog timer.

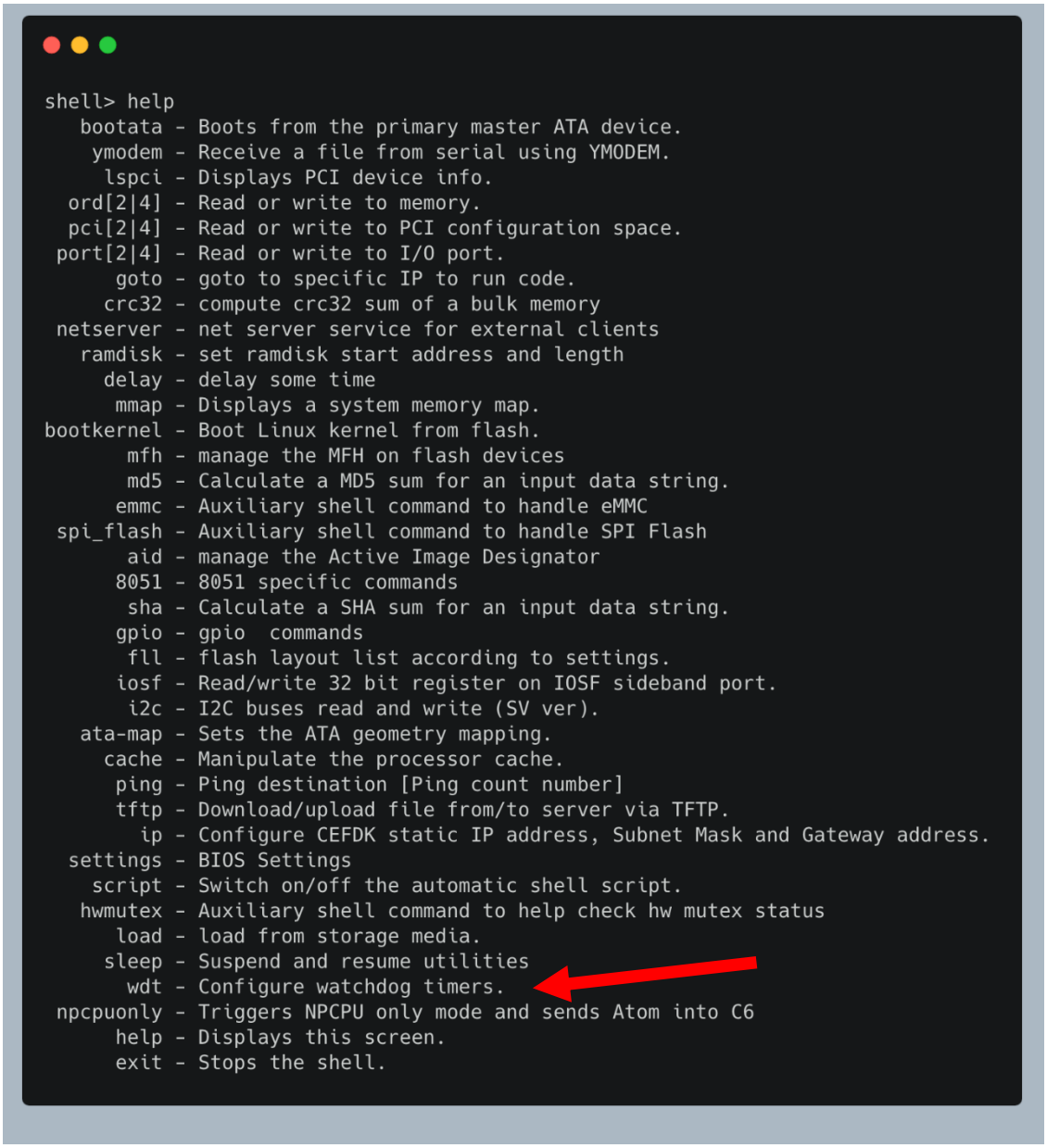
What Is It?

A watchdog timer (WDT) is a hardware or software timer that will typically cause a system reset if it is not periodically "kicked" or "fed". Generally, on embedded systems the main programming is in some type of a loop. Each iteration of this loop the WDT is kicked to ensure that the system is not reset. Since it is common for embedded systems to be unmonitored or unmanaged and potentially control critical systems, the developers want to quickly resolve issues with the system. A WDT provides a way to quickly reset the system to try and mitigate issues that may cause the system to hang. It is also common that a backup firmware image or known good state is stored on a flash chip. If a WDT is kicked too many times consecutively, the known good image will be loaded in place of the currently running image. This hopefully resets the system back to a known good state as quickly as possible. It is important to keep this in mind when researching embedded systems while debugging since it is possible to unknowingly switch running firmware.

Disabling Hardware WDTs

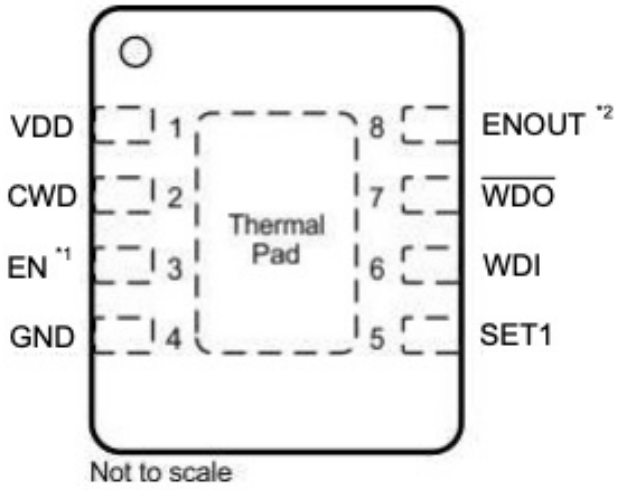
There are three general approaches to disabling a hardware watchdog timer which will be explained in order of increasing difficulty.

If it is possible to gain an interactive interface with the device, such as UART, there is sometimes a command to disable a hardware watchdog timer. This is designed for developers during testing and is common on debug ports. This command usually interacts with a pin on the watchdog timer which simply maintains whether the timer is enabled or disabled. This pin is sometimes referred to as the "SET" pin. This is by far the simplest way to disable a hardware watchdog timer, if you have access to such a command. Below is an example of a command list from a real embedded device. Notice there is a "wdt" command. This can be used to disable the timer on this device.



Another option is to consider that the software already running on the device must have the functionality to deal with the WDT. Depending on how this code is implemented it is possible to use these functions to your advantage. If you can identify which functions are used to kick the watchdog and they are exported, an additional program can be written to execute the kicking code at the will of the researcher. A great example of this can be found in this Delta Controls blog.

Lastly, if the above two methods are not possible, a hardware approach can be deployed in order to disable a hardware watchdog timer. WDTs typically have a pin which handles the incoming clock and a pin which handles sending a reset to the processor, WDI and WDO respectively.



Generally, when the software "kicks the dog" it sends a falling edge signal to the WDI pin before the timeout. It is sometimes possible to override this by connecting an external input source to the WDI pin. This could be something like a function generator. By using the generator to produce the falling edge at an interval before the timeout, it is possible to kick the dog and prevent a reset. In contrast, when the timer expires it sends a reset command to the microcontroller or processor over the WDO pin. Another method to prevent this is to disrupt this communication, by lifting the WDO pin off the board or pulling this pin high constantly. Both of these methods run a higher risk of damaging the device, are more complex and can be more invasive, however may produce the desired results.

Disabling Software WDTs

Sometimes developers will add an additional watchdog timer in software. This might be done to add more checks or to take different actions beyond just a hard reset to mitigate issues. Since these are done in software the first trick is to find the section of code that is handling the counter. If you are lucky enough to have symbols this may not be too difficult using function names and strings. Without symbols, a helpful pattern to look for is a function being called very often between operations or every time within each thread or main loop.

Similarly, for the first method explained for disabling hardware watchdog timers, sometimes it is possible to find a function which simply disables the software watchdog timer. Like before, this is used by developers during development and is not always removed. If this is found and the functions are exported by the programing, it is once again possible to simply call this function to disable the software WDT from an external program.

If there are no functions that have been created to disable the software watchdog timer, once the section of code being used to control it has been identified, a technique called binary patching can potentially be used to disable the timer. Binary patching modifies a binary file typically by using a hex editor in order to edit code flow. With binary patching it is usually best to make the smallest code change possible since you do not want to have unintended results. With watchdog timers there is typically a function which is decrementing a counter that represents the timer. A good place to binary patch is to modify only the immediate value being used to decrement the timer. For example, if the timer is being reduced by "3" each time binary patch the value to be "0". This will render the watchdog ineffective. For a real-world example of this please see the Delta blog mentioned earlier.

Watchdog timers are a great addition to embedded systems and serve a very important purpose. Unfortunately, this also can make a researcher's job more difficult. By employing some of these techniques it may be possible to circumvent these timers and continue to work on these devices.

McAfee and the McAfee logo are trademarks or registered trademarks of McAfee, LLC or its subsidiaries in the US and other countries. Other marks and brands may be claimed as the property of others. Copyright © 2020 McAfee, LLC. 4431_0320
MARCH 2020