High-Level JTAG Exploits

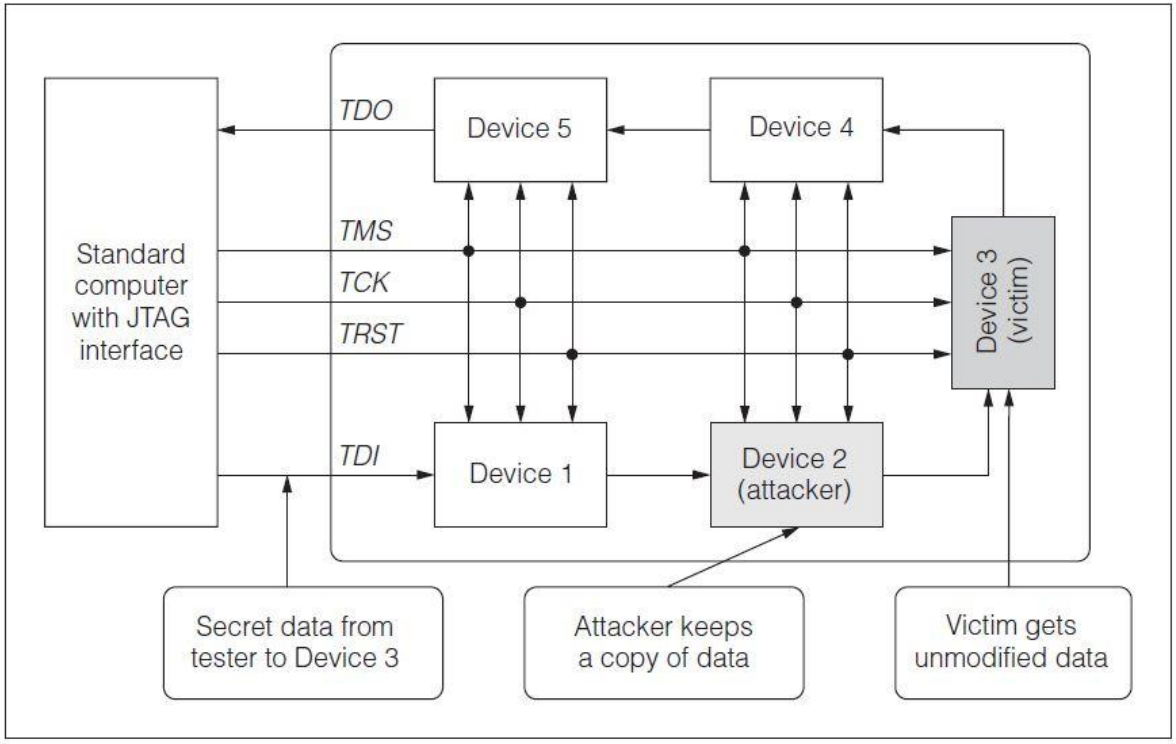
- Sniff TDI/TDO signals
- Modify TDI/TDO signals
- Control TMS and TCK signals

Sniff TDI/TDO Signals

- Sniffing attacks require the insertion of an attacker JTAG module. Using this attacking module and normal JTAG function, an attacker can reveal secrets, test vectors, and test responses
- One attack using JTAG is to sniff the TDI or TDO lines of a victim chip. In this attack, a module in the JTAG chain, either before or after the victim module, will intercept the data being sent or coming from the victim module. The attack module acts differently to the bypass mode. To the JTAG system, it seems to act like it is bypassing data from TDI to TDO, but it will actually be storing that data for the attacker to look at. Data from the victim device may be test-vectors, test responses, or secret keys.

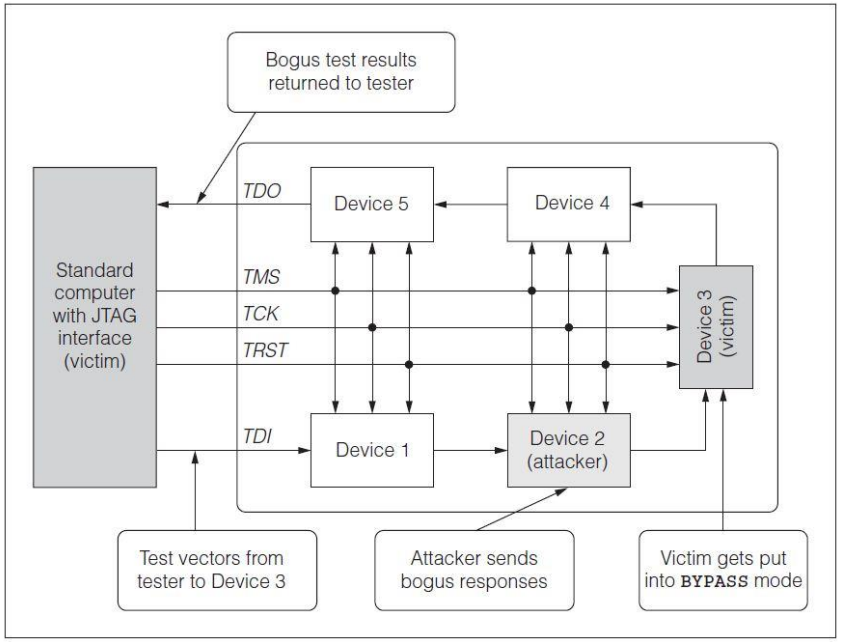
Sniff TDI/TDO Signals

- Used to intercept secrets being sent to or from a chip
- Preceding or chip after victim chip behaves differently during bypass to intercept message



Modify TDI/TDO Signals

- Can modify Test Vectors and Test Responses
- Can be used to fake correct or false tests
- Attacker can either be upstream or downstream of victim based on attack



Control TMS and TCK Signals

- For many exploits, TMS and TCK signals need to be controlled
- Attacker needs to be able to overpower The signal sent by TAP
- Attacking device needs to be able to force TMS and TCK above or below logic threshold voltage
- Can be done by combining lines to make a more powerful driver or using multiple attackers to overcome TMS and TCK signals

Examples of JTAG attacks

- JTAG attacks are very common and can be done with little technical equipment and knowledge. To successfully perform this attack, you need a handful of resistors, some wire, a serial port, female connection, a soldering iron, and a few diodes. All these parts are easily purchased online from companies like DigiKey.

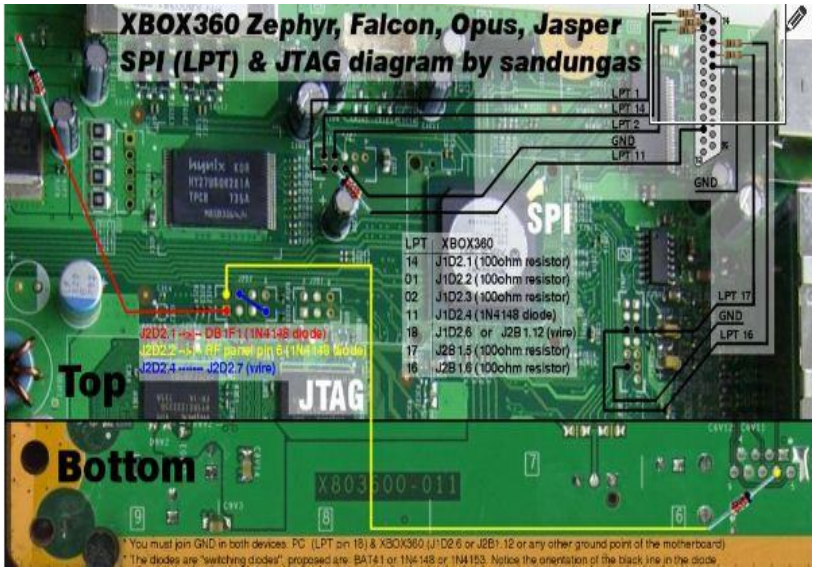
- A popular exploit of JTAG that has been exposed recently is using the JTAG port in an XBOX 360 to gain privileges that Microsoft never intended.

Xbox 360 Exploit

- By using a cable connected from the JTAG port to ones computer, the nand memory of the XBOX can be dumped to the computer, and flashed. Dumping the memory allows the attacker to gain the key values used by the XBOX and the nand memory can also be flashed in a similar attack to allow the attacker to unlock secure features of the XBOX.
- Using this type of attack allows people to add extra hard drives to their XBOX, run homebrew code on the XBOX, rip games to the hard drive, unlock demos, and cheat in games. The most publicized use of this attack was to allow players to modify the game, Call of Duty: Modern Warfare 2. This attack shows both weaknesses of JTAG discussed earlier.

Xbox 360 Exploit

- Used to override Microsoft security features
- Allows homebrew code to be run, installation of HD, game modification, ripping of games
- JTAG is used to extract secret keys needed to perform exploits and to change programming
- This attack exposes secrets inside the hardware, and allows the hardware to be updated and loaded with non-licensed software.



Security Options

- Buffers in the JTAG Chain
- JTAG system connected in “Star” pattern instead of being chained(Separate TMS and TCK)
- Encryption/Authentication for JTAG use
  - Most of the research in JTAG security would be classified under this
  - Although it would provide much better protection, like all security hardware, increases cost and space.

Security Options

- Buffers and star patterns provide minimal security but make many attacks impossible to do are not expensive in terms of cost, size, and timing.
  - prevent the TMS and TCK signals from being tampered with. Although many attacks require that these signals be controlled, this minimal security does not prevent all attacks.
- The buffer and star security options also require the designer to add complexity, and area overhead.
- Encryption/Authentication provides a much more secure JTAG system, however adds a lot of complexity, area overhead, and cost.

Star connections

- **Central Control Point:** In a star connection, all JTAG devices are connected to a central hub (usually a test controller or a central monitoring point). This setup allows the controller to manage and control access to each device individually, making it easier to monitor and restrict access to sensitive devices.
- **Isolation:** Star connections isolate devices from each other, preventing an attacker from accessing other devices on the JTAG chain by compromising a single device. If one connection is tampered with, it doesn’t expose the entire chain, unlike a traditional daisy-chain JTAG configuration.

Buffers

- **Signal Integrity and Isolation:** Buffers act as intermediary circuits that maintain signal strength and integrity across longer distances, preventing issues like signal degradation. This is especially useful when securing JTAG connections, as weak signals can introduce noise and make tampering harder to detect.
- **Controlled Access:** Buffers can be used to enable or disable signals selectively, controlling when and which parts of the JTAG connection are active. For example, a buffer can lock certain lines unless authenticated, blocking unauthorized attempts to communicate with devices on the JTAG interface.

Buffers

- **Tamper Detection and Response:** Advanced buffers can detect signal anomalies that might indicate tampering. If an unusual signal is detected, the buffer can alert the central controller or disconnect the device from the JTAG interface, blocking further access.

Encryption/Authentication

- JTAGs secured by public/private key encrpytion methods, with challenge/response authentication, and ways of limiting use based on permission levels.
- All these options greatly increase area overhead and because of multi-cycle authentication/encryption methods add to the complexity of testing through the JTAG port.

Challenge, Response

- A very basic way of determining whether a module on a JTAG system is trusted is to do authentication through a challenge, response set-up.
- In this type of authentication, the tester will provide a module with a challenge. The module will then hash the challenge with a unique value inside the module and provide a response.
- In this type of authentication, the tester must know the unique value inside the module and be able to calculate the response themselves.
- If the expected response matches the response through the JTAG port, the tester can be confident that the module is what they expected.

Challenge, Response

- Requires PUF or randomly burned fuses
- Requires Set_Challenge and Get_Response instructions in JTAG implementation
- A Challenge input is given to the JTAG module, and the module will hash this with the value of it's fuses to create the response
- Only a known, trusted module will give a correct response
- So, can be determined if modules are trusted or not

Public/Private Key Authentication

- Public/Private key cryptography can be used to secure a JTAG system while reducing the number of keys the JTAG system is required to save.
- Using a designated third party to sign authentication certificates for updaters or testers of modules
- The JTAG system can check the authenticity of these certificates using a single public key.
- This would greatly improve security of the JTAG system while only adding some area overhead and an initial delay during authentication.

Public/Private Key Authentication

- The goal of this security option is to make sure that any tester or updater that is using the JTAG port is trusted.
- Common JTAG exploits are to update the design in re-programmable devices to make them perform differently or add functionality.
- With this security option, it would require anyone trying to update any of the JTAG modules to be a trusted updater, hopefully preventing malicious updates.
- Requiring a certificate of authentication would also not allow other JTAG attacks that require an attacker to run tests or other instructions through the JTAG port to extract secrets.

Public/Private Key Authentication

- This type of JTAG security is very feasible but does add a large area overhead.
- It is feasible because only a single key would have to be stored to decrypt any certificates, but the extra hardware to do the decryption may add more area overhead than a designer wants.
- Although this option does not interrupt the timing of the JTAG system, or require any changes to the JTAG protocol, it does provide an initial delay during authentication of a certificate.

Public/Private Key Authentication

- Tester/Updater is required to have a certificate of authentication signed by a designated third party.
- Authenticators public key is known to JTAG system
- Using the known public key, the JTAG system can decrypt the certificate and determine whether the tester/updater is trusted
- Trusted testers/updaters are allowed access to JTAG system, un-trusted are blocked

User Permissions

- User permissions allows for different levels of control through the JTAG port.
- May only allow testing for one type of user and updating to another.
- This concept needs to be combined with an authentication process.

- Using permission levels to control what testers or updaters have access to is not a whole security system.

User Permissions

- Using permission levels would still require some sort of authentication hardware to be added to the JTAG system.
- But the idea behind user permissions is to leave the JTAG port open and functional, but limit what can be done through the JTAG port through a set of permissions.
- These permissions can be defined by individual modules and this system would seem to resemble User permissions on a computer or what Active Directory does for Windows machines.

User Permissions

- A user permission level, i, allows them access to instructions with a level less than i
- Requires extra hardware to authenticate user and set permission level, and to save settings for what each permission level can and cannot do
- Ex. In memory, a permission level is saved for each module in the JTAG system. When that module is trying to be accessed, the saved level is compared to the current permission level

Removal/Destruction of JTAG

- Removal or destruction of the JTAG hardware may seem extreme, but it is the only way to guarantee that JTAG exploits can not be developed. A problem with building modules that do not include JTAG support is how to test them. Designers can include a BIST (Built-in Self Test) routine to check for errors. However, BIST routines have been proven to not provide as much test coverage, and lack the ability to isolate a problem.

Removal/Destruction of JTAG

- Some companies have started to implement a way to disable the functionality of the JTAG hardware after it has been used for testing.
- In this design, the JTAG hardware is implemented with security fuses that can be blown by the tester once the chip has been determined as working.
- Using security fuses to disable JTAG has also allowed companies to only disable certain features of JTAG.
- In these designs, a set of fuses can be blown that may only disable the ability to scan out internal registers, but leave the ability to run tests.

Removal/Destruction of JTAG

- To completely defend against JTAG attacks, one thought is to remove the JTAG hardware all together
  - Does not leave a way to in-field test
  - Can use BIST for testing

- Similarly to removal of JTAG, some companies use security fuses to disable JTAG before the hardware leaves the factory
  - Can implement different levels of disabled JTAG use