NutriSafe

Sicherheit in der Lebensmittelproduktion und -logistik durch die Distributed-Ledger-Technologie

Giesecke+Devrient

NutriSafe Toolkit

– DLT-Sicherheitsanalysen –

Verbesserung der Sicherheit von Sicheren Elementen (DLT Hardware Wallets) gegen physikalische Angriffe

Oscar Guillén Hernández – Michael Lamla – Thomas Furtner

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

Bundesministerium Landwirtschaft, Regionen und Tourismus

SIFO.de

KIRAS Sicherheitsforschung

FFG Forschung wirkt.

Dieses Dokument ist Bestandteil im **NutriSafe Toolkit**:

nutrisafe.de/toolkit

In einer Kooperation zwischen Deutschland und Österreich forschen Universitäten, Unternehmen und Behörden daran, die Lebensmittelproduktion sowie deren Logistik unter Nutzung von Distributed-Ledger-Technologie sicherer zu machen.

Das Projekt NutriSafe wird auf Österreichischer Seite innerhalb des Sicherheitsforschungs-Förderprogramms KIRAS durch das Bundesministerium für Landwirtschaft, Regionen und Tourismus (BMLRT) gefördert (Projektnummer: 867015). Auf Deutscher Seite wird das Projekt innerhalb des Programms Forschung für die zivile Sicherheit vom Bundesministerium für Bildung und Forschung (BMBF) gefördert (FKZ 13N15070 bis 13N15076).

nutrisafe.de | nutrisafe.at

Verbesserung der Sicherheit von Sicheren Elementen (DLT Hardware Wallets) gegen physikalische Angriffe

Oscar Guillén Hernández[1], Michael Lamla[1], Thomas Furtner[1]

[1]Giesecke+Devrient Mobile Security GmbH

München 2021

Giesecke+Devrient Mobile Security GmbH

Prinzregentenstr. 159

81677 München

Dieses Werk ist lizenziert unter einer
Creative Commons Namensnennung - Keine Bearbeitungen 4.0 International Lizenz
(http://creativecommons.org/licenses/by-nd/4.0/).

Inhalt

1 Verbesserung der Sicherheit von Sicheren Elementen (DLT Hardware-Wallets) gegen physikalische Angriffe

1.1 DLT Hardware-Wallets

Ein Wallet enthält Informationen zur Identität eines Benutzers. Zugangsrechte zum Distributed Ledger werden an Hand von dieser Information bestimmt. Außerdem werden dadurch alle Daten, die in einer DLT (Distributed Ledger Technologie) stehen, durch eine Transaktion in die DLT gebracht. Transaktionen werden authentisiert, in dem ein Benutzer jede Transaktion mittels eines Private Key (privater Schlüssel) signiert.

Eine Wallet kann vollständig in Software implementiert sein. Allerdings können Implementierung bei denen wichtige Daten extern gespeichert werden, insbesondere die Private Keys eines DLT Benutzers, die Sicherheit erhöhen. Reine Software Implementierungen sind nicht robust genug gegen Software Angriffen, die diese geheime Information extrahieren oder korrumpieren können.

Eine Hardware-Wallet hingegen ist eine Lösungen, in der die Wallet auf einem externen Gerät, wie z.B. einer Smartcard realisiert wird. Diese Hardware-Wallet wird nur während der Verwendung mit einem Computer verbunden und ist somit die meiste Zeit offline und damit nicht für klassische Hacker erreichbar.

Auf dieser Hardware-Wallet sind unter anderem die Private Keys des DLT-Benutzers gespeichert. Diese Keys bleiben in dem externen Gerät und sind deswegen nicht bedroht, falls der Rechner Opfer eines Software Angriffs wird. Die Information einer Transaktion wird innerhalb der Hardware-Wallet signiert. Danach wird die signierte Information an den Rechner übermittelt. Der Benutzer kann mit einer PIN die Hardware-Wallet schützen. Daher werden für die Signierung einer Transaktion eine PIN (etwas, dass der Benutzer weiß) und die Hardware-Wallet (etwas, dass der Benutzer besitzt) gebraucht.

Die Sicherheit einer Hardware-Wallet liegt darin, dass die in dem Gerät gespeicherten geheimen Informationen nicht extrahierbar sein dürfen. Zu diesem Zweck muss die Hardware-Wallet selbst resistent gegen Software Angriffe sein aber auch gegen sogenannte physikalische Angriffe, wie Seitenkanal-Analysen und Fehlerangriffe.

1.2 Fehlerangriffe

Fehlerangriffe sind eine Klasse von physikalischen Angriffen auf Rechnersysteme und ihre Komponenten, in der ein Angreifer versucht *aktiv* ein Gerät zu stören mit dem Ziel Sicherheitsüberprüfungen umgehen zu können oder kryptographische Berechnungen gezielt zu manipulieren.

Im Gegensatz zu *passiven* Angriffs-Methoden, wie Seitenkanal-Analysen, bei denen der Angreifer Informationen während des normalen Ablaufs eines Prozesses sammelt und analysiert, versucht der Angreifer mit Fehlerangriffen das Rechensystem in einen kontrollierten

fehlerhaften Zustand zu bringen, um daraus einen Vorteil zu gewinnen. Durch kurzzeitige Veränderung von Betriebsparametern auf Hardwareebene, wie zum Beispiel der Pegel in der Versorgungspannung, werden die Berechnungen gezielt beeinflusst und damit Fehler induziert.

Durch erfolgreiche Störungen können beispielsweise Sicherheitsabfragen im Software-Code übersprungen werden. Wird zum Beispiel bei einer If-Else-Anweisung, die die Zugriffsrechte auf eine Datei kontrolliert, ein Fehler induziert und dadurch das Ergebnis manipuliert, kann ein nicht berechtigter Nutzer Zugriff auf diese Datei bekommen.

Bei Angriffen von kryptographischen Verfahren wird gezielt in einem ersten Schritt ein Fehler in einen Berechnungsschritt induziert. In einem zweiten Schritt kann aus den erzeugten fehlerhaften Ergebnissen mittels mathematischer Analyse, auf die Geheimnisse (z.B. Schlüssel wie Private Keys) zurückgerechnet werden.

Daher stellen Fehlerangriffe auf eingebettete Systeme, unabhängig von dessen logischem Sicherheitsniveau eine Gefahr für die in ihnen gespeicherten geheimen Informationen dar.

Ein Angreifer hat mehrere Angriffspfade zur Verfügung, um sich Zugriff auf die geheimen Informationen zu beschaffen. Dies sind zum Beispiel:

- Fault Analysis: Kryptographische Schlüssel mit Hilfe von mathematischen und statistischen Verfahren abzuleiten (zum Beispiel, Differential Fault Analysis auf symmetrische oder asymmetrische Kryptographische Algorithmen).
- (Wieder-) Einschaltung von Debug- und Programmier-Schnittstellen.
- Kombinierte Software / Hardware Angriffe: Reguläre Software Kommandos durch die Kombination von unerlaubte Parametern und einen gezielten Hardware Angriff so zu verändern, dass geheime Informationen ausgelesen werden können (zum Beispiel, Software Gegenmaßnahmen gegen Buffer Overflows können mittels Fehlerangriffen überbrückt werden, um Informationen zu bekommen, die in einem anderen Speicherbereich liegen als das Kommando im Prinzip dürfte).

Es gibt ein breites Spektrum von Methoden, um Fehlerangriffe durchführen zu können. Die Quellen können intern oder extern sein. Eine Klasse stellt die Einbringung bösartiger Software Attacke dar, indem man die Konfiguration eines Systems modifiziert (zwei bekannte Beispiele sind ClkScrew [1] und Plundervolt [2]). Eine andere Klasse von Methoden ist dadurch gekennzeichnet, dass sie Fehler mittels physikalischer Geräte generiert und direkt auf der Hardware-Ebene induziert.

Auch wenn software-basierte Fehlerangriffe möglichenfalls aus der Entfernung (z.B. übers Internet) durchgeführt werden können, sind sie oft auch einfach zu beheben, indem man beispielsweise Zugriffsrechte entsprechend konfiguriert. Zusätzlich stellen sie eher ein Risiko für System-on-Chip Geräte dar, die von mehreren Nutzern gleichzeitig genutzt werden können als für einfachere (sicherere) Mikrokontroller-basierte Geräte, bei denen die Programme begrenze Aufgaben erfüllen.

Hardware-basierte Fehlerangriffe hingegen, benötigen einerseits physikalischen Zugriff auf das Gerät, andererseits ist es schwieriger sich gegen diese Angriffe zu schützen. Gleichzeitig betrifft es ein breiteres Spektrum von Geräten, die dieser Gefahr ausgesetzt sind. So stellt der Schutz gegen derartige Angriffe eine wichtige Kategorie für Hardware-Wallets dar, um zu verhindern, dass diese bei Diebstahl so manipuliert werden können, dass auf gespeicherte, schützenswerte Inhalte (wie z.B. der Private Key) zugegriffen oder manipuliert werden können.

Der Angreifer kann verschiedene Methoden von hardware-basierten Fehlerangriffen einsetzen, die unterschiedliche physikalische Parameter benutzen. In dem folgenden Text werden wir uns nur mit hardware-basierten Fehlerangriffen beschäftigen. Tabelle 1 zeigt einen Vergleich von üblichen hardware-basierten Methoden.

Tabelle 1 – Physikalisch Fehlerangriff Methoden (unvollständig)

Methode	Zeitliche Genauigkeit	Örtliche Genauigkeit	Zugriff
Takt Störungen	Sehr hoch	-	Nicht-invasiv
Spannung Spikes	Mittel/Hoch	-	Nicht-invasiv
Alphastrahlung	Niedrig	Sehr niedrig	Nicht-invasiv
EM Strahlung	Mittel/Hoch	Niedrig/Mittel	Nicht-invasiv / Semi-invasiv
Laserimpulsen	Hoch	Hoch	Semi-invasiv

Die zeitliche Genauigkeit beschreibt die Fähigkeit eine Operation präzise stören zu können. Takt-Störungen (auch als Glitches bekannt) zeigen die höchste zeitliche Genauigkeit bei Prozessoren, die mit einem externen Takt laufen. Bei allen anderen Methoden muss ein Trigger-Signal generiert werden. Dafür werden spezielle Geräte eingesetzt. Entscheidende Faktoren sind die Verzögerung zwischen Triggersignal, Auslösezeitpunkt und Dauer des Fehlereffekts. Störungen mittels Laserimpulse zeigen hierbei die besten zeitlichen Eigenschaften.

Örtliche Genauigkeit bedeutet mit welcher Granularität die Fehlereffekte eine Schaltung beeinflussen können. Taktstörungen und Spannungsspikes erzeugen globale Effekte über den ganzen Chip. Alphastrahlungen von einer ionisierenden Strahlenquelle können Bits in dem Schaltkreis zufällig kippen, durch Abschirmung kann der Einfluss auf bestimmte Bereiche begrenzt werden. Elektromagnetische Strahlungen lassen sie auf einen Teil des Chips fokussieren, indem man elektromagnetische Sonden mit kleinen Spulen verwendet um eine Spannungsvariation in begrenzen Teilen des Chips zu erzeugen. Laserimpulse lassen sich mit Hilfe von speziellen Objektiven genau fokussieren und bieten damit die beste örtliche Genauigkeit. Mit ihnen kann man auf Register- oder abhängig von der Technologie auch in der Transistor-Ebene Fehler erzeugen. Wegen der örtlichen und zeitlichen Genauigkeit ist ein Laserangriff daher die meistverbreitete Methode, um Hardware-Sicherheitsuntersuchungen durchzuführen.

Auch wenn für Fehlerangriffe stets physischer Zugriff auf das Hardware-Wallet vorhanden sein muss, muss man differenzieren, ob der Angreifer die Störungen nur über externe Kontakte einbringen kann, oder ob der Angreifer die Fähigkeit besitzt den Chip zu präparieren und somit Zugriff auf interne Signale erlangt. Für den Zugriff auf interne Signale gibt es drei Stufen: Nicht-invasiv, Semi-invasiv und Invasiv. Bei nicht-invasiven Angriffen wird der Chip nicht präpariert und der Angreifer benutzt nur die bereits verfügbaren Pins für die Attacken. Bei semi-invasiven Attacken wird der Chip einfach präpariert, sodass das Silizium des Chips aufgedeckt wird. Dies kann mittels chemischer oder mechanischer Verfahren erfolgen. Invasive Angriffe gehen noch ein Schritt weiter. Bei diesen werden Schichten des Chips teilweise entfernt, um Zugriff auf einzelne Leiterbahnen zu gewinnen. Um invasive Attacke durchführen zu können sind hoch spezialisierte Geräte nötig.

1.3 Fehlerangriffe mittels Laserimpulse

Fokussiertes Laserlicht erzeugt über den photoelektrischen Effekt kurze Variationen in der Ladung von Transistoren in einem Chip. Damit können die Spannungspegel in einem örtlichen begrenzen Bereich beeinflusst werden. Die Effekte hängen davon ab, wo man in dem Layout des Integrierter Schaltkreises angreift und welche Komponenten dadurch beeinflusst werden. Der Angreifer kann dadurch im Prinzip statische Daten modifizieren in:

- Speicherzellen im flüchtigen Arbeitsspeicher (RAM)
- Speicherzellen im nichtflüchtigen Dauerspeicher (NVM)
- CPU Register
- Sonderfunktions-Register

Zusätzlich kann er auch gezielt dynamische Daten während der Berechnungen stören in:

- Prozessorkernen
- Hardwarebeschleunigern
- Adressdecodern im flüchtigen und nichtflüchtigen Speicher

Im Zusammenhang mit Hardware-Wallets sind Störungen auf statistischen sowie dynamischen Daten relevant für ihre Sicherheit. Geheiminformationen wie Private Keys und PINs werden im nichtflüchtigem Dauerspeicher gespeichert. Bei Störungen von dynamischen Berechnungen, könnte der Angreifer die PIN-Vergleichsroutine überspringen oder die Berechnung so beeinflussen, dass eine falsche PIN als richtig erkannt wird.

Laserangriffe können auf der Vorderseite oder Rückseite eines integrierten Schaltkreises durchgeführt werden. Lichtquellen mit verschiedener Wellenlänge sind für jede Seite ausnutzbar. Um auf der Vorderseite des Chips anzugreifen, werden meist Laser im sichtbaren Lichtspektrum benutzt. Nahe Infrarot Laser (NIR) werden im Gegensatz dazu verwendet, um Angriffe über die Siliziumrückseite auszuführen, weil Silizium die Eigenschaft hat, für Licht im nahen- und mittleren Infrarotbereich transparent zu sein. Angriffe auf die Rückseite eines Chips haben den Vorteil für den Angreifer, dass die Transistoren direkt getroffen werden, weil keine Leiterbahnen zwischen der Lichtquelle und den Transistoren im Weg stehen. Abbildung 1 zeigt eine vereinfachte Ausschnittdarstellung eines integrierten Schaltkreises.

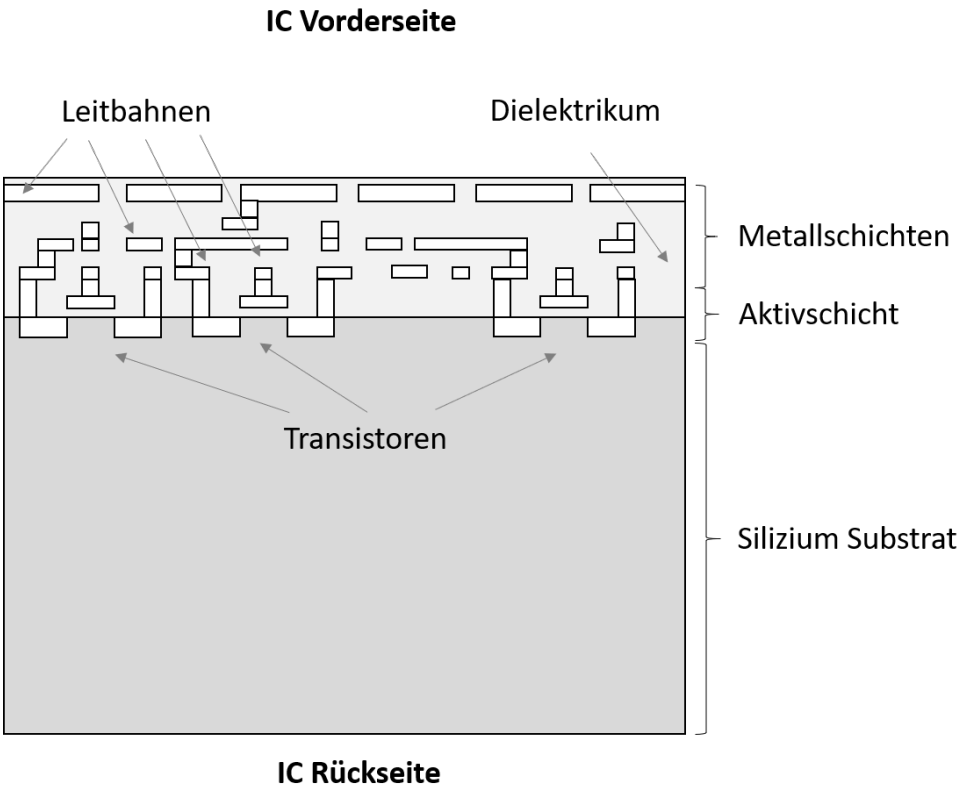

Abbildung 1 - Vereinfachte Darstellung eines CMOS integrierten Schaltkreis (Ausschnitt)

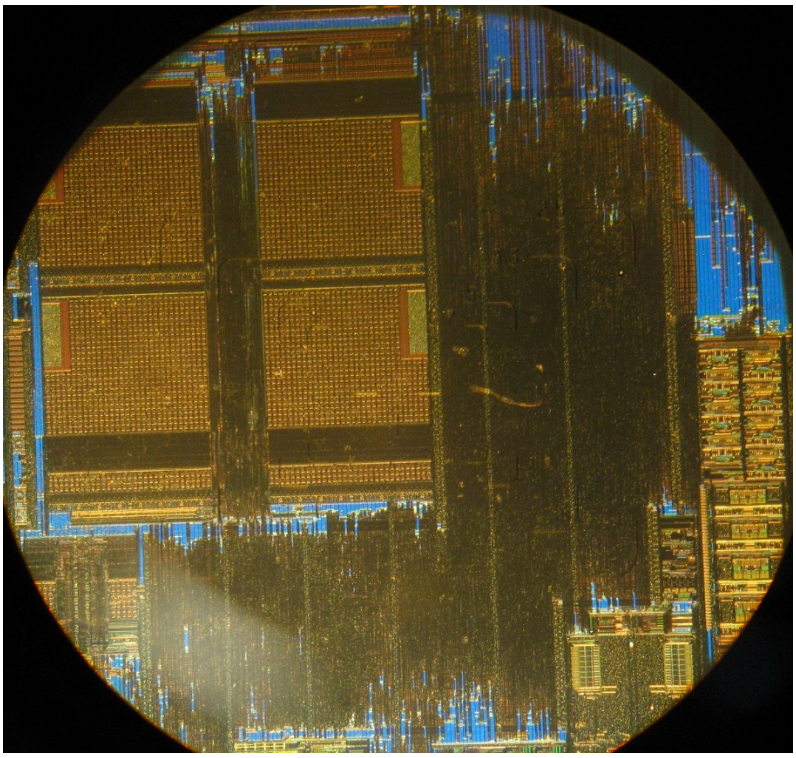

Abbildung 2 - Vorderseite eines Hardware-Wallet-Mikrocontrollers durch ein optisches Mikroskop

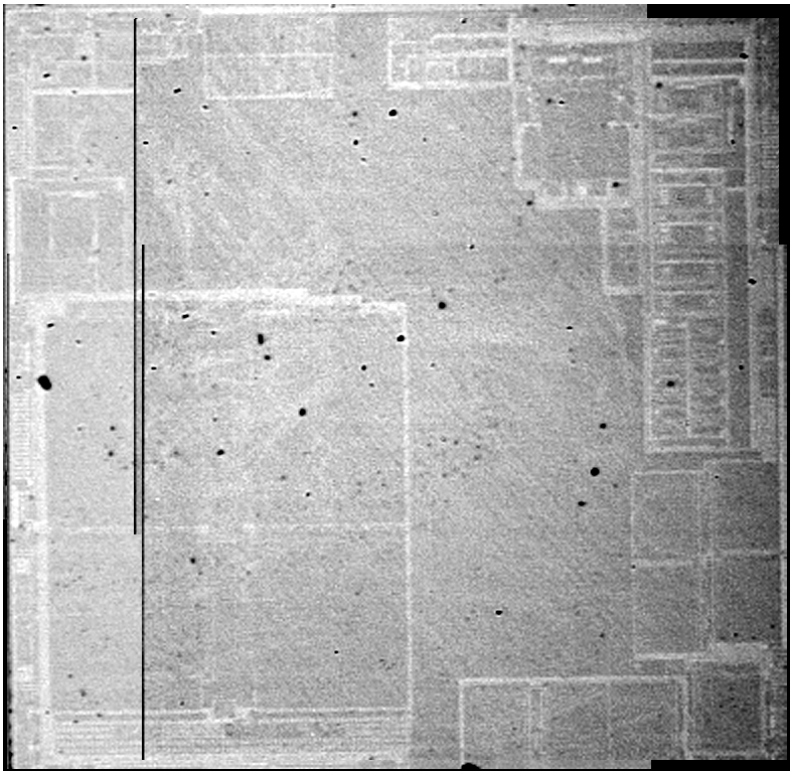

Abbildung 3 - Rückseite eines Hardware-Wallet-Mikrokontrollers mittels einer Infrarotkamera (Bild-Stitch von verschiedenen Teilbildern)

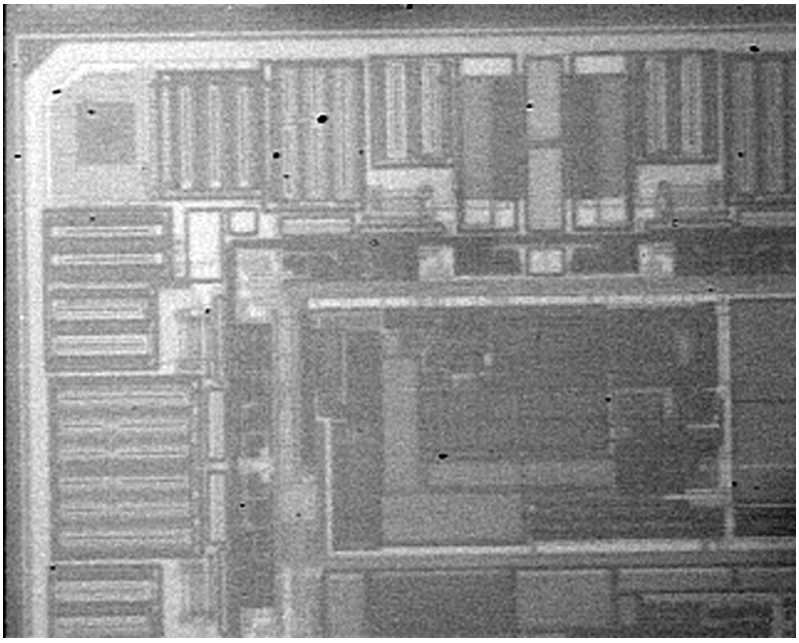

Abbildung 4 - Rückseite eines Hardware-Wallet-Mikrokontrollers mittels einer Infrarotkamera (Zoom auf einen Bereich mit verschiedenen Analogkomponenten)

Um die gewünschten Angriffseffekte zu bekommen muss ein Angreifer mehrere Parameter optimieren. Für Laser Fehlerangriffe sind die wichtigsten Parameter:

- Platzierung auf dem Chip (X-Y Koordinaten)
- Wellenlänge (UV, sichtbares Licht, Nahinfrarot, Infrarot)
- Fokusgröße des Laserstrahls auf dem Angriffsobjekt

- Energie des Laserstrahls
- Laser Pulsbreite (zeitliche Länge der Störung)
- Zeitpunkt der Störung (oft relativ zu einem Trigger Zeitpunkt)
- Wiederholrate des Laserpulses
- Anzahl der Laserpulse

Da der Suchraum sehr groß werden kann, werden normalerweise zu Beginn programmierbare Samples eingesetzt, um ein erstes Feedback zu bekommen. Hardware-Wallets die einen nicht sicheren Mikrokontroller verwenden, sind daher einfacher zu charakterisieren als sichere Mikrokontroller. Diese sicheren Mikrokontroller sind nicht frei verkäuflich und daher für Angreifer nur schwer zugänglich. Typischerweise muss ein Hersteller von Hardware-Wallets einen Geheimhaltungsvertrag unterschrieben, bevor er Zugriff auf Tools und Datenblätter für sichere Mikrokontroller erhält.

1.4 Gegenmaßnahmen gegen Fehlerangriffe

Wenngleich bei Fehlerangriffen die Störungen physikalisch in den einzelnen Hardwarekomponenten erzeugt werden, so lassen sich diese oft auch als Fehler auf höheren Abstraktionsebenen beschreiben. Um sich gegen diese Art von Angriffen zu schützen gibt es verschiedene Strategien, die auf den unterschiedlichen Abstraktionsebenen implementiert werden können. Tabelle 2 listet typische Beispiele für Gegenmaßnahmen gegen Fehlerangriffe auf.

Tabelle 2 - Gegenmaßnahme gegen Fehlerangriffe auf verschiedenen Abstraktion-Ebenen

Ebene	Gegenmaßnahme
Transistoren / Leitbahnen	Lichtsensoren, Gitter,
Logische Gatter / einfache Schaltungen	Filtern, Timing-Fehler-Detektoren, Differentiale Logik, Temperatur Sensoren, Dummy Warteschleifen, Takt Jitter, Rauschgeneratoren
Komponenten	Mehrfachregister, Bus Verwürfelung, Mehrkernprozessoren
Arithmetische	Fehlerkorrekturverfahren, Zyklische Redundanzprüfungen

Algorithmisch	Mehrfachberechnungen, Kontrollfluss-Integrität, Differentielle Berechnung, Dummy Operationen
Protokolle	Integritätsprüfung, Schlüssel Behandlung

Der Hardware-Wallet-Entwickler hat im Prinzip vier Strategien zur Verfügung, um sich zu schützen. Im Rahmen einer Defense-in-Depth Strategie sind das:

- Physikalische Veränderungen detektieren (z.B. durch den Einsatz von Lichtdetektoren)
- Physikalische Veränderungen erschweren (z.B. durch Redundanz)
- Effekte auf höheren Abstraktionsebenen detektieren (z.B. durch Fehlererkennungs-verfahren)
- Auf Angriffe reagieren (z.B. Funktionen blockieren oder Daten löschen)

In Hardware-Wallets können möglicherweise mehrere Gegenmaßnahmen in verschiedenen Ebenen implementiert werden. Die Sicherheit eines Produktes wird durch die Kombination aus Hardware- und Softwaregegenmaßnahmen bestimmt. Insgesamt gilt somit, dass je sicherer die Hardware ist, desto weniger Aufwand muss in Software-Gegenmaßnahmen investiert werden und andersrum. Abbildung 5 zeigt ein Beispiel Layout eines Mikrokontrollers für Hardware-Wallets.

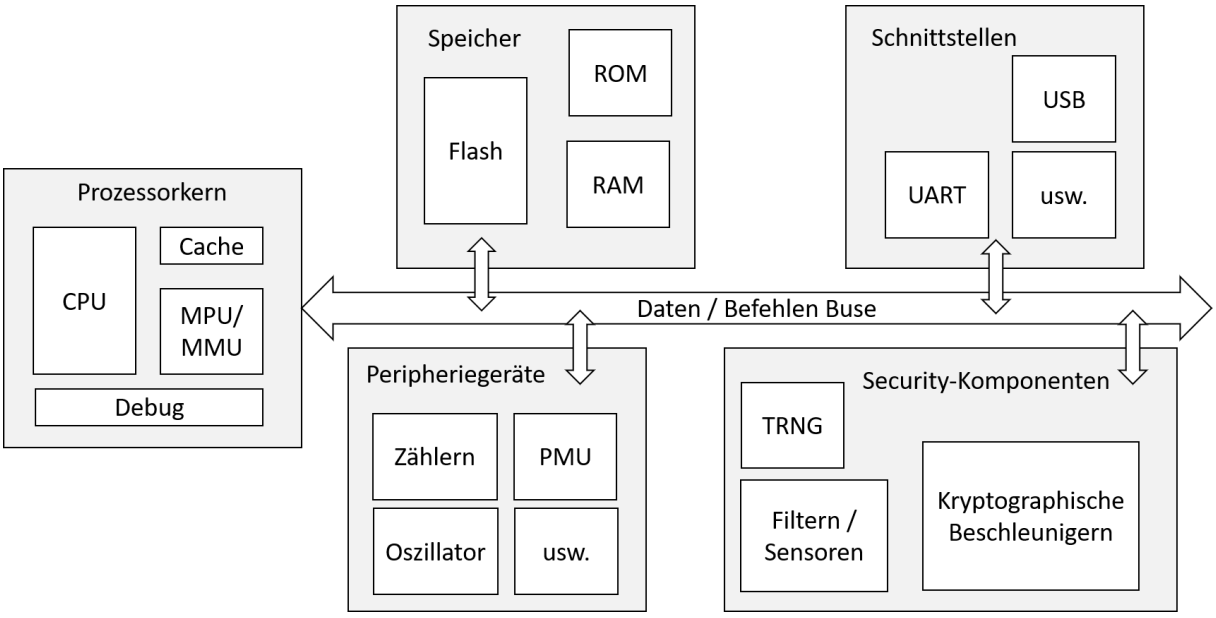

Abbildung 5 - Komponenten eines Mikrokontrollers für Hardware-Wallets

1.5 Praktische Untersuchungen

Da Hardware-Wallets nicht standarisiert sind, wird jeder Hersteller seine eigene Lösung konzipieren. Es gibt daher Hardware-Wallets die mit einfachen Mikrokontrollern ausgestattet sind. Diese Hardware-Wallets bieten zwar dem Benutzer die Möglichkeit seine Private Keys extern zu speichern, trotzdem sind sie aber gegen Fehlerangriffe nicht resistent. Für praktische Untersuchungen wurden Mikrokontroller (Sichere Elemente) im Smartcard Format verwendet. Sichere Elemente mit einem sicherem Betriebssystem sind seit Jahren für Bezahlanwendungen (z.B. Kreditkarte) bereits auf Sicherheit gegen Software- und Hardwareangriffe optimiert. Dies ist bei normalen Hardware-Wallets noch nicht der Fall.

Ein speziell entwickeltes Betriebssystem wurde benutzt, um Information über den Zustand interner Register zu erhalten. Aus Gründen der Geheimhaltung können Information über den Hersteller und Chips nicht bekanntgegeben werden. Deshalb werden im Folgendem die Ergebnisse von den Untersuchungen, sowie die implementierten Gegenmaßnahmen, in allgemeiner Form präsentiert.

Die Mikrokontroller wurden mittels Laserstrahlung bzgl. ihrer Angreifbarkeit untersucht. Dazu wurden sie in erster Linie mechanisch so präpariert, dass die Rückseite des Chips zugänglich wurde. Die Chips wurden danach in einem spezialangefertigten Halter platziert. Der Halter mit einer Schnittstelle ist auf einem X-Y Tisch montiert, um mit dem Lichtstrahl mikrometergenau darüber scannen zu können. Die Teststation ist mit einer Lichtquelle ausgestattet, die über ein weites Spektrum, vom sichtbaren bis zum nahen infrarot Licht, verfügt. Ein Objektiv mit einer 50-fachen Vergrößerung, welches speziell für diesen Spektralbereich optimiert ist, ermöglicht die optische Darstellung des Chiplayouts von der Rückseite. Der Laser wird über einen Pulsgenerator getriggert. Laser und X-Y Tisch werden über einen Rechner gesteuert. Abbildung 6 stellt den Laser plus Objektiv dar und Abbildung 7 zeigt ein Blockdiagram mit den verschiedenen Komponenten der Teststation.

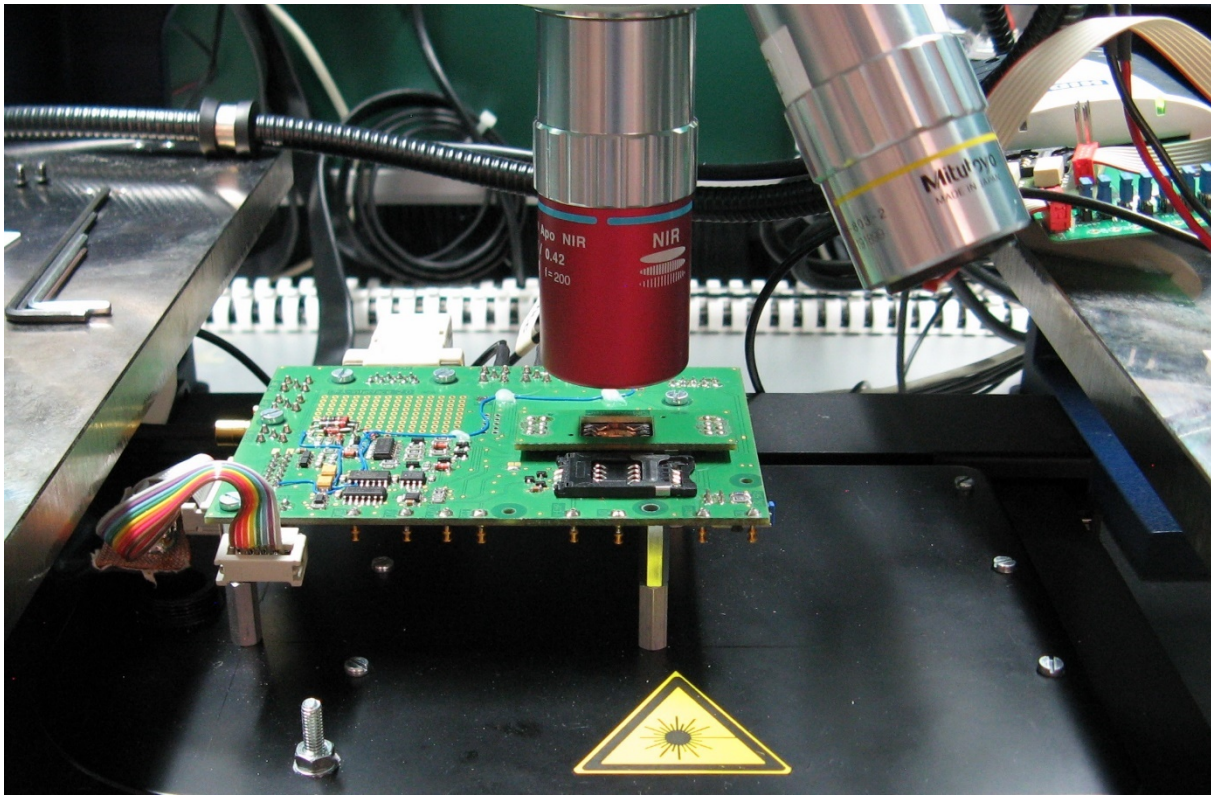

Abbildung 6 – Der Laser wird mittels Objektiv fokussiert

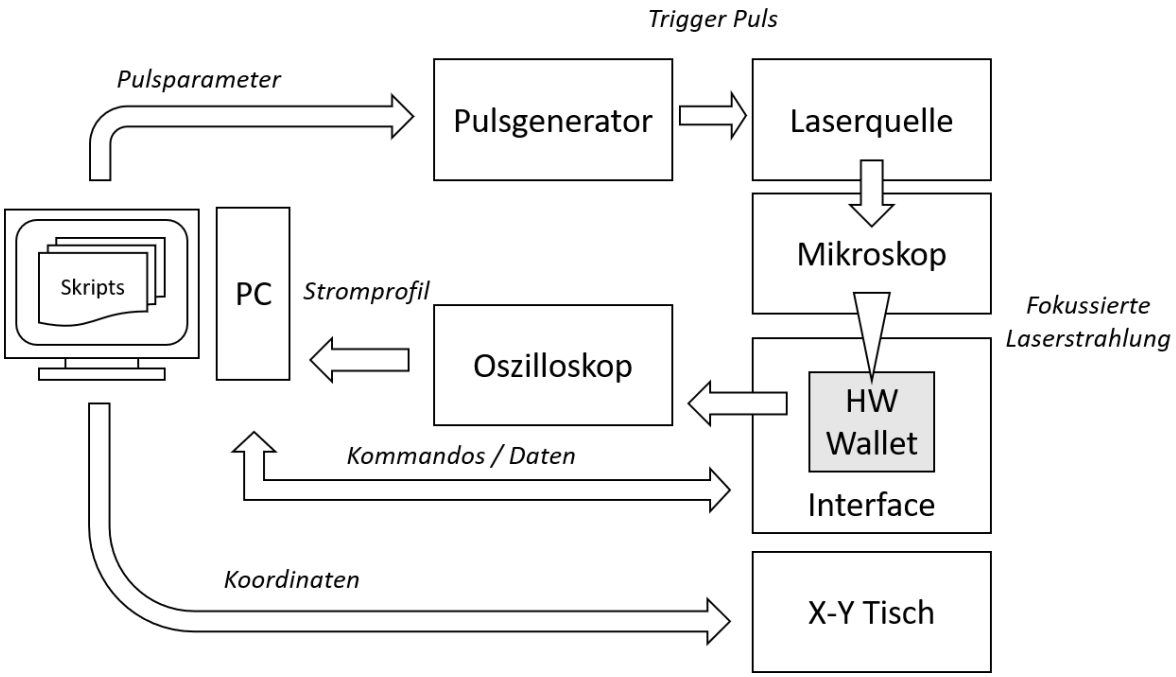

Abbildung 7 - Blockdiagram der Komponenten der Teststation

1.5.1 Speichergrenze - Checks Umgehen

Die in der Hardware-Wallet gespeicherten Informationen dürfen für einen Angreifer nicht lesbar sein. Ein üblicher Angriffsvektor ist der Versuch, mehr Inhalt aus dem Speicher auszulesen, indem man die Prüfung von Speichergrenzen mit Fehlerangriffen attackiert. Die erstellten Speicher Dumps können danach offline analysiert werden. Im Jahr 2019 wurde ein Angriff dieser Art auf einer kommerziellen Hardware-Wallet publiziert [3]. Speicher Dumps konnten durch EM Fehlerangriffe im USB Software-Stack durchgeführt werden.

Bei unseren Tests wurde ein Kommando angegriffen, das für das Senden statischer Informationen verantwortlich ist. Dieses Kommando sendet nach seiner Ausführung eine feste Anzahl von Bytes zurück. Die Information ist ursprünglich im nichtflüchtigen Speicher gespeichert, wird jedoch während des Bootvorgangs in den RAM-Speicher kopiert. Das Kommando holt die Daten aus dem Arbeitsspeicher und sendet diese über eine Kommunikationsschnittstelle an den Benutzer zurück. Software-Prüfungen stellen sicher, dass die Daten aus einem korrekten Adressraum gelesen werden und dass nur die zulässige Anzahl von Bytes zurückgesendet werden.

Ziel dieser Tests war es, sicherzustellen, dass keine weiteren Informationen extrahiert werden können, als das Kommando zurückschicken darf. Als erster Schritt eines Angriffs wurde das komplette Chip-Layout mit einer Schrittweite von 5x5um und in einem durch die Analyse von dem Stromprofil definiertem Zeitfenstern gescannt. Ein Bereich in der Digital-Logik, in dem der Chip gestört werden konnte, wurde damit gefunden. Die Koordinaten für die nachfolgenden Scans wurden auf diesen angreifbaren Bereich beschränkt. Die Laserleistung musste dabei ständig angepasst werden, weil sonst einige Chips auf Grund der hohen Intensität des Lasers zerstört worden wären. Es wurde entschieden, mit niedriger Laserintensität zu arbeiten. Die Wahrscheinlichkeit einen erfolgreichen Fehler zu provozieren verringert sich dadurch, dafür wird aber der Chip nicht zerstört (im Gegensatz zu einer höheren Intensitäten, die mehr gewollte Fehler erzeugt, aber den Chip zerstören kann). Abbildung 8 zeigt die Ergebnisse eines Testlaufs mit optimalen Fehlerinjektions-Parametern.

Die Effekte wurden in vier Kategorien sortiert: Kein Effekt, Mutes (auch als „keine Antwort“ bezeichnet), Tickles, und Heurekas. Kein Effekt bedeutet, dass das Programm zum Signieren von DLT-Transaktionen auf dem Mikrokontroller der Hardware-Wallet normal ausgeführt wird. Mutes oder keine Antwort bedeutet, dass der Chip durch den Angriff in einen Zustand gekommen ist, in dem er innerhalb eines bestimmten Zeitfenster keine Antwort gesendet hat. Tickles sind Ergebnisse die fehlerhaft sind aber die Sicherheit der Gegenmaßnahmen nicht kompromittieren. Ein Beispiel ist, wenn weniger Daten als erwartet geschickt werden, oder wenn das Testkommando mit einer unerwarteten (aber sicheren) Antwort reagiert. Heurekas sind Fälle bei denen die Angriffe funktioniert haben und das Kommando mehr Information von der Hardware-Wallet zurückschickt, als es sollte.

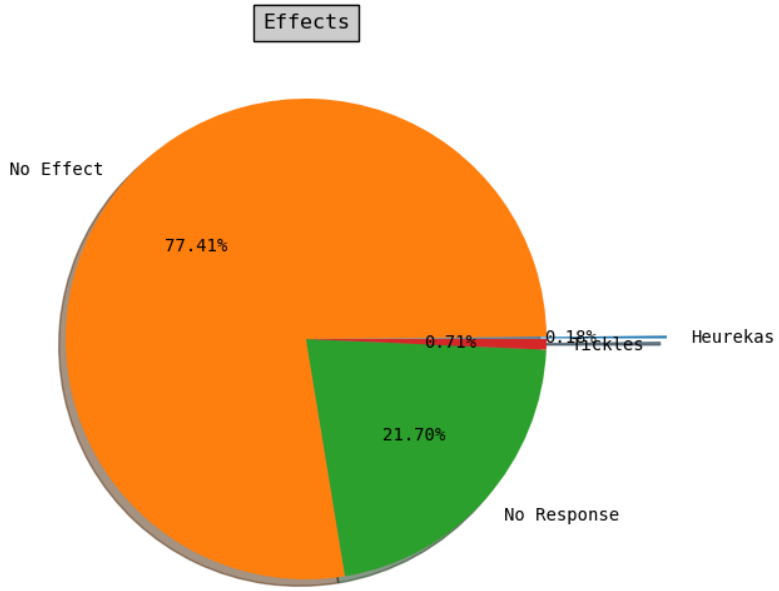

Abbildung 8 – Effekte bei Speicher Checks Szenario

Das Zeitfenster für die Tests wird in Abbildung 9 gezeigt. Darin ist es ersichtlich, dass das Programm zu verschiedenen Zeitpunkten gestört werden konnte. Die Heurekas Zeit wurden kurz vor Ende der Ausführung des Kommandos gefunden.

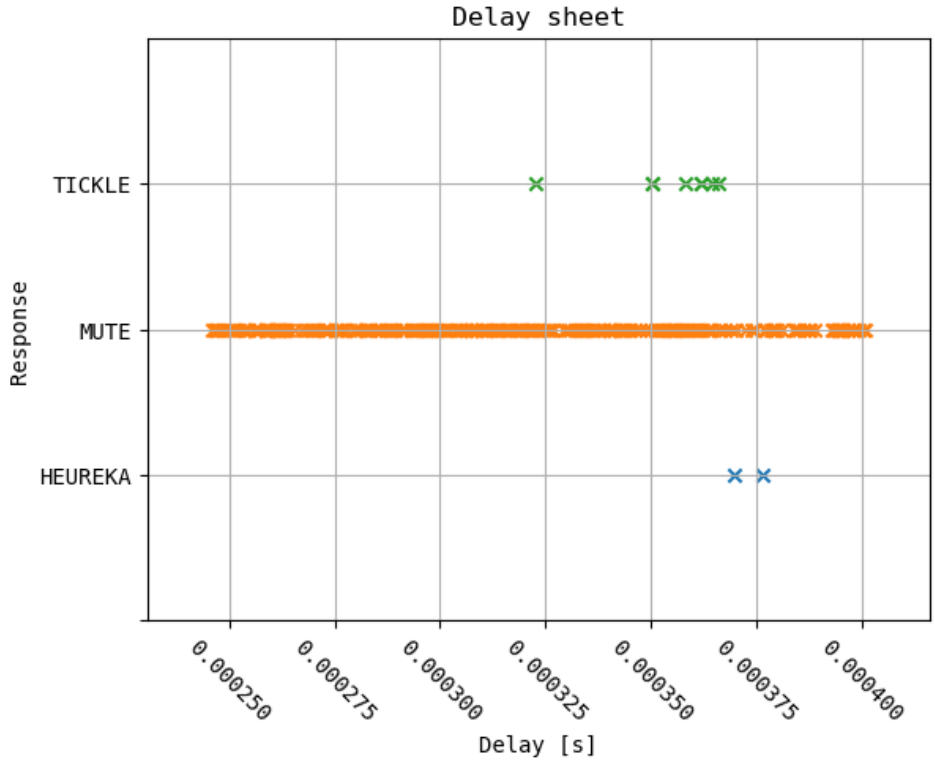

Abbildung 9 - Zeitfenster mit Effekten für den Tests

Diese Ergebnisse lassen sich reproduzieren und die Erfolgsquote kann erhöht werden, indem man das Zeitfenster und die X-Y Koordinaten weiter begrenzt. Auch wenn sich herausstellte, dass die während dieser Tests extrahierten Informationen nicht geheim waren, zeigten die

Ergebnisse aber, dass trotzdem weitere Gegenmaßnahmen in die untersuchte Routine eingebaut werden müssen.

1.5.2 PIN-Code Vergleich Bypass

Die PIN schützt die Hardware-Wallet vor unerwünschtem fremdem Zugriff. Wird die Vergleichsroutine durch Fehlerangriffe umgegangen, könnte ein Angreifer die Hardware-Wallet benutzen, um Daten für die DLT zu signieren.

Für den Tests wurde die PIN-Code Vergleichsroutine angegriffen. Diese Routine übernimmt die vom Benutzer eingegebene PIN, leitet mittels kryptographischem Verfahren einen Hash Wert ab und vergleicht diesen mit dem in der Hardware-Wallet gespeicherten Wert. Die geschützten Funktionen der Hardware-Wallet (z.B. signieren von DLT-Transaktionen)können erst benutzt werden, wenn die PIN Eingabe zuvor erfolgreich war.

Ein Zähler wird benutzt, um die Anzahl der PIN-Eingabe Versuche zu begrenzen. Dieser Zähler wird in jedem Versuch inkrementiert und im nichtflüchtigem Speicher abgelegt. Der Zähler wird nur dann zurückgesetzt, wenn die richtige PIN eingegeben wurde.

In den ersten Tests wurde der PIN Zähler deaktiviert und die PIN-Code Vergleich Routine untersucht. Das Ziel war es, einen Benutzer zu authentisieren, obwohl die eingegebene PIN falsch war. Durch Auswertung verschiedener Stromkurven konnte ein Zeitfenster identifiziert werden, in dem Angriffe erfolgsversprechend sind. Mindestens drei korrekt platzierte Angriffe wären erforderlich, um die Vergleichsroutine zu umgehen. Die Ergebnisse zeigten, dass dies in einer praktikablen Zeit nicht möglich war.

Da der Angreifer nicht die Möglichkeit hat den PIN Zähler zu deaktivieren müsste er in der Lage sein, den Angriff mit höchster Präzision durchzuführen. Ein Vorteil für den Angreifer wäre es, wenn er den PIN Zähler zurücksetzen könnte. Daher konzentrierten sich die nächsten Tests darauf die PIN Zähler Erhöhung zu vermeiden. Da der Code prüft, ob der Zähler korrekt inkrementiert wurde, führte der Versuch nur dazu, dass die Karte stumm geschaltet wurde. Es wurde auch beobachtet, dass in einigen Fällen die Anzahl der verbleibenden Versuche verringert werden konnte. Dies stellt eigentlich keine Sicherheitslücke dar, da es für den Angreifer kein Vorteil sondern einen Nachteil ist. Tatsächlich würde dieser Effekt dazu führen, dass die Karte früher ungültig gemacht würde, wodurch sich die Anzahl der Versuche, die dem Angreifer zur Verfügung ständen, verringern würde.

1.6 Fazit

Hardware-Wallets werden empfohlen, um die Geheimnisse (Private Keys) die für die erfolgreiche Signierung einer DLT-Transaktion nötig sind, auch im Falle der Kompromittierung eines PCs sicher aufzubewahren. Allerdings müssen Hardware-Wallets auch vor spezifischen Hardware-Attacken wie Fehlerangriffe geschützt werden. Hierfür sollten Sichere Elemente verwendet werden, die Sicherheitsmechanismen in verschiedenen Abstraktionsebenen bereitstellen. Die Software, die auf diesen Geräten läuft, sollte ebenfalls sicher programmiert werden, um zu gewährleisten, dass die Werte auch dann sicher bleiben, wenn der Angreifer in

der Lage wäre, Hardware-Gegenmaßnahmen zu umgehen. Die Interaktion zwischen Hardware und Software muss daher durch praktische Tests bewertet werden. Nur so kann gewährleistet werden, dass die Gegenmaßnahmen erwartungsgemäß funktionieren.

2 Literatur

[1] Tang, Adrian, Simha Sethumadhavan, and Salvatore Stolfo. "{CLKSCREW}: exposing the perils of security-oblivious energy management." *26th {USENIX} Security Symposium ({USENIX} Security 17)*. 2017.

[2] Murdock, Kit, et al. "Plundervolt: Software-based fault injection attacks against Intel SGX." *2020 IEEE Symposium on Security and Privacy (SP)*. 2020.

[3] O'Flynn Colin, "Glitching Trezor using EMFI Through The Enclosure", online: https://colinoflynn.com/2019/03/glitching-trezor-using-emfi-through-the-enclosure/