In Arbeit 10.15.4 + Clover 5109 auf MSI X99A 5820K I Trotz Umfassenden EFI ist Installation von OSX Nicht Möglich I Probleme mit boot.efi

Beitrag von „griven“ vom 20. April 2020, 16:36

Nein, nein und nochmals nein!

Alles was Du im Clover Configurator eintragen kannst bzgl. des SMBIOS hat keinerlei Einfluss auf Dein Problem denn der Fehler tritt bereits auf bevor das SMBISO von macOS überhaupt erkannt/verarbeitet wird. Bitte bitte mach Dich erstmal schlau bevor Du weiter wild in der Weltgeschichte herumspekulierst. Die Stichworte auf die es ankommt (Sorry ich habe aktuell nicht die Zeit das genau zu erklären da auf der Arbeit) sind Kernel Adress Space Layer Randomization, MemoryMap und die Funktion der Aptio Firmware als solcher. Das Problem hat nichts aber auch rein gar nichts mit dem RAM in Deinem Rechner zu tun oder damit wie der an macOS weitergegeben wird. Die unterschielichen AptioFixe (AptioMemoryFix, OSXAptioFix, FWRuntimeServices inkl. OCQuirks) arbeiten im Tanzbereich der Firmware also irgendwo zwischen Ende initialisierung des Rechners durch das UEFI und Start des Betriebssystems durch den Bootloader. Im Falle von macOS will die boot.efi (der Bootloader von macOS) den Betriebsystemkern (Kernel) gerne an einen zusammenhängenden Speicherbereich unterhalb der 4GB Grenze entpacken und von dort starten und genau das schlägt in Deinem Fall fehl.

Zum Verständnis die Firmware des Rechners (UEFI/BIOS) schreibt beim initialisieren des Rechners Informationen zu der im Rechner vorhandenen Hardware in den Speicher das sind in erster Linie Informationen darüber unter welchen Adressen welche Geräte erreichbar sind usw. die Menge der Informationen schwankt mit der Menge der verfügbaren PCI Lanes (daher sind X Serie Boards auch öfter von dem Problem bertoffen als Z Serie Boards) und mit der Menge der verbauten Steckkarten denn je mehr Steckkarten ums so mehr landet auch im Speicher. Die Firmware geht dabei mehr oder weniger verschwenderisch mit den Ressourcen um denn ausser für macOS ist es nicht sonderlich wichtig das im Bereich unterhalb von 4GB möglichst unfragmentierter freier Speicher verfügbar ist (und Windows Board sollen bekanntlich auch kein macOS booten). Die diversen Memoryfixe nehmen sich nun diesen Problems an indem sie analysieren wie der Speicher belegt ist und versuchen die Fragmentierung des Speichers aufzulösen (um zu erklären wie das genau funktioniert fehlt mir zum einen die Zeit und zum anderen auch das Verständnis das können andere sicher besser als ich spielt hier aber auch nur eine untergeordnete Rolle). Je nach Generation des Fixes gelingt das mehr oder weniger gut. Der AptioMemoryFix und die FWRuntimeServices (quasi der "Nachfolger" des

Memoryfixes) haben hier die ausgefeiltesten Strategien auf Lager und können weit mehr als "nur" Speicheradressen virtuell umbiegen die einfachen Fixes beschränken sich auf das umbiegen der Adressen. Du siehst also das alles hat nix mit dem SMBIOS oder dem RAM im Rechner zu tun sondern eben damit wie die Firmware mit den gegebenen Ressourcen umgeht und genau deshalb macht es auch keinen Sinn ASUS mit MSI Mainboards zu vergleichen denn das ist so als vergleiche man Äpfel mit Birnen. Schlimmer noch es kann sein das ein und das selbe Mainboard bei User A läuft und bei User B ums verrecken nicht was dann unter anderem daran liegen kann das beide unterschiedlich viele Erweiterungskarten einsetzen oder eben einfach auch nur andere Karten verwenden (verschiedene Grafikkarten etc.).

Jetzt noch ein paar Worte zum Slide Wert und was der eigentlich macht. Wie ich eingangs schon schrieb solltest Du Dich mal schlau machen was eigentlich gemeint ist wenn von KASLR die Rede ist denn der Slide Wert steht damit im direkten Zusammenhang. Gültige Werte für Slide sind Ganzzahl Werte zwischen 1 und 256 wobei der Wert bei jedem Start des Systems per Zufallsgenerator ausgewürfelt wird. Dieser Wert gibt einen Versatz ein sogenanntes Offset an von dem ausgehend die Bootloader versucht den Kernel im Speicher zu installieren. Ein Beispiel der Slide Wert 0 würde dazu führen das ausgehend von der Startadresse 0x0000000000 ein Offset von 0 hinzuaddiert würde sprich boot.efi würde versuchen den Kernel an der Basisadresse zu installieren und wirft einen Fehler aus wenn das nicht erfolgreich ist sowas in der Art:

Code

```
1. Error allocating 0x1197b pages at 0x0000000017a80000 alloc type 2
2. Couldn't allocate runtime area
```

Gibt man keinen Slide Wert vor dann wird bei jedem Start ein anderer verwendet und es kann sein das dabei zufällig einer getroffen wird der zu einem ausreichend großen freien Bereich passt und das System durchbootet (passiert ja Bei Dir zuweilen auch siehe Dein Screenshot 2 -> Übrigens keine MemoryPanic sondern vermutlich eher wegbrechender USB Support aber das ist ein anderes Thema). So nun habe ich mehr geschrieben und Zeit investiert als ich eigentlich gerade habe aber vielleicht hilft Dir das das richtige Verständnis für die Sache zu entwickeln und an den richtigen Stellen zu schrauben (-> OCQuirks -> OcQuirks.plist und hier insbesondere die Punkte DevirtualiseMmio, AvoidRuntimeDefrag, ProvideCustomSlide und SetupVirtualMap)...