7. Übung zur VL Betriebs- und Kommunikationssysteme

Tutor: Thomas Tegethoff

Bearbeiter: Etienne Jentzsch, Carola Bothe

1a)

Directory Traversal

Trickst ein schwaches Programm aus, sodass es Zugang zu einer geschützten/inneren Datei gewährt indem es den Verzeichnisbaum hoch und dann runter zur Passwort Datei geht. Als Verteidigungsmaßnahme sichert/verschlüsselt man die Passwortdateien und erlaubt web Servern keinen vollen Zugriff auf Dateisysteme. (VL Folie 7.29, Mitschrift)

Buffer Overflow

Bezeichnet man, wenn ein Programm Daten über die Grenzen eines Puffers auf benachbarten Speicherplatz schreibt. Dadurch kann bösartiger Code auf diesen benachbarten Speicherplatz geschrieben und später ausgeführt werden. (https://en.wikipedia.org/wiki/Buffer\_overflow)

Trapdoor/Backdoor

Ursprünglich als Tool zum debuggen und testen gedacht, erlaubt es mittels Hintertüren den Zugang zu einer Datei/einem Programm ohne den üblichen Sicherheitsprozess zu durchlaufen. Die Tür wird durch ein Stück Software implementiert (dazu werden im Negativfall z.B. Geräte beim Transport abgefangen) und kann vom Betriebssystem nur schwer kontrolliert werden. (VL Folie 7.35, Mitschrift)

Logic Bomb

Gehört zu den ältesten Schädlingen und ist ebenfalls Teil eines Programms. Sie 'explodiert' wie der Name sagt, wenn eine bestimmte (logisch) Bedingung erfüllt ist. Die Folgen der Explosion können veränderte oder gelöschte Daten, ein Abstürzen des Systems oder anderes sein. (VL Folie 7.36, https://de.wikipedia.org/wiki/Logikbombe)

Trojan Horse

Angeblich nützliches Programm, das die gewünschte Funktion ausführt aber zusätzlich eine bösartige Funktion hat. Beispiel wäre eine Taschenlampen Anwendung, die nebenbei Kontaktdaten übermittelt, was man hier zumindest mit der Ablehnung von Zugriffsrechten im Zaum halten kann. (VL Folie 7.37, Mitschrift)

Virus

Software, die nach einem Auslöser andere Software modifiziert und neben dem schädlichen Code auch Code zur Selbstreplikation enthält. Viren haben heutzutage lange Inkubationszeiten, damit sie auch auf etwaigen Backups vertreten sind, und die Fähigkeit zur Mutation, sodass Virenprogramme an ihre Grenzen stoßen. (VL Folie 7.40, Mitschrift)

Worm

Sind wie Zombies unabhängig vom Gastbetriebssystem und sobald sie einmal aktiviert wurden, replizieren sie sich wie Viren, wobei sie aktiv nach weiteren Verbreitungsmöglichkeiten auf andere System und Netzwerke suchen. In Gegensatz zum Virus findet die Verbreitung ohne die Infektion von fremden Dateien oder Bootsektoren statt.

(VL-Folie7.45, https://web.archive.org/web/20101124010540/http://service1.symantec.com/SUPPORT/nav.nsf/pfdocs/1999041209131106)

Bot (aka Zombie, Drone)

Ein Programm, das heimlich andere am Internet hängende Geräte übernimmt, von denen dann die Attacke gestartet wird. Besonders im Zeitalter des Internets stellen sie daher eine Bedrohung dar und man kann die Bots selten zu ihrem Ersteller zurückverfolgen. (VL Folie 7.48, Mitschrift)

Rootkit

Sind Programme, die auf dem System Root Zugang gewähren, indem sie die Standardfunktionalität verändern. Damit wird Zugang zu allen Funktionen und Diensten des Betriebssystems gewährt und der Angreifer hat die komplette Kontrolle des Systems. (VL Folie 7.49)

b)

BIOS steht für Basic Input Output System und UEFI für Unified Extensible Firmware Interface. Bei beiden handelt es sich um read only Code (ist also nicht beschreibbar), der die Schnittstelle zwischen Firmware, Computer Komponenten und dem OS bildet also beim Systemstart u. A. den Speicher prüft und Treiber lädt. Während BIOS das booten aller bootloader (Treiber) erlaubt, erlaubt UEFI aus Sicherheitsgründen nur signifizierte. Das UEFI ist im Gegensatz zum BIOS ein eigenes kleines Betriebssystem, sodass etwa Updates direkt über das UEFI geladen und installiert werden können. Mit UEFI kann man nur 64-Bit Systeme booten und außerdem mehr primäre Partitionen einer Festplatte einrichten und größere Speichermedien verarbeiten. Wir finden UEFI besser, da es durch seine Standards für mehr Sicherheit sorgt und leichter Updates ermöglicht und dazu eine benutzerfreundliche Oberfläche bietet.

(VL Folie 7.6 & 7.8, Mitschrift, http://praxistipps.chip.de/bios-oder-uefi-das-sind-die-unterschiede\_36099)

c)

10 gut verstandene Begriffe:

Rings, scheduling algorithms, Process, swapping, fragmentation, partitioning, virtual memory, hierarchical file system, priorety inheritance, preemptive

Weniger gut verstandene Begriffe:

I/O Buffering, Daemons, dispatcher, countermeasures,

2a) +Vorteil / - Nachteil

(Named) Pipe oder auch FIFO-Pipes:

**+** Im Gegensatz zu Unbenannten Pipes können Benannte Pipes auch zur Kommunikation zwischen Prozessen benutzt werden, die nicht miteinander verwandt sind. Einer Datei wird also ein gewisser Name zugeteilt und jeder Prozess, der diesen Namen kennt, kann darüber die Verbindung zur Pipe und damit zu anderen Prozessen herstellen.

**-** Für benannte Pipes müssen Zugriffsrechte verteilt werden.

https://de.wikipedia.org/wiki/Interprozesskommunikation#Shared\_Memory

Memory Mapped File:

Ein Memory-Mapped-File(Speicher-abgebildete Datei) enthält den Inhalt einer Datei im virtuellen Speicher. + Diese Zuordnung zwischen einer Datei und einem Speicherplatz ermöglicht es einer Anwendung, einschließlich mehrerer Prozesse, die Datei durch Lesen und Schreiben direkt im Speicher zu ändern. Es gibt zwei Arten:

Persisted (beharrte) Memory-Mapped-Files

Sind Dateien, die mit einer Quelldatei auf einem Datenträger verknüpft sind. Wenn der letzte Vorgang mit der Datei beendet ist, werden die Daten in der Quelldatei auf dem Datenträger gespeichert. + Diese Speicher-abgebildeten Dateien eignen sich für die Arbeit mit extrem großen Quelldateien.

Non-Persisted (nicht beharrte) Memory-Mapped-Files

Dateien, die nicht mit einer Datei auf einem Datenträger verknüpft sind. Wenn der letzte Prozess mit der Datei beendet ist, gehen die Daten verloren und die Datei wird durch Garbage Collection zurückgefordert. Diese Dateien eignen sich für die Erstellung von Shared Memory für Inter-Process Communications (IPC).

+ Speicher-abgebildete Dateien können über mehrere Prozesse freigegeben werden

+ Prozesse können dieselbe Speicher-abgebildete Datei zuordnen, indem sie einen gemeinsamen Namen verwenden, der dem Prozess zugeordnet ist, der die Datei erstellt hat

- Um mit einer Speicher-abgebildeten Datei zu arbeiten, muss man eine Ansicht der gesamten Speicher-abgebildeten Datei oder eines Teils davon erstellen. Sie können auch mehrere Ansichten zu demselben Teil der Speicher-abgebildeten Datei erstellen, wodurch ein gleichzeitiger Speicher erzeugt wird. Für zwei Ansichten, die gleichzeitig bleiben sollen, müssen sie aus derselben Speicher-abgebildeten Datei erstellt werden.

Es können auch mehrere Ansichten erforderlich sein, wenn die Datei größer ist als die Größe des logischen Speicherplatzes der Anwendung für die Speicherzuordnung (2 GB auf einem 32-Bit-Computer).

Es gibt zwei Arten von Ansichten: Stream-Zugriffsansicht und zufällige Zugriffsansicht. Verwenden Sie Stream-Zugriffsansichten für den sequentiellen Zugriff auf eine Datei; Dies wird für nicht beharrte Dateien und IPC empfohlen. Zufällige Zugriffsansichten werden für die Arbeit mit beharrten Dateien bevorzugt.

Speicher-abgebildete Dateien werden über den Speicher-Manager des Betriebssystems abgerufen, so dass die Datei automatisch in eine Anzahl von Seiten partitioniert und nach Bedarf zugegriffen wird. Du musst das Gedächtnismanagement nicht selbst behandeln.

Die folgende Abbildung zeigt, wie mehrere Prozesse gleichzeitig mehrere und überlappende Ansichten auf dieselbe Speicher-abgebildete Datei haben können.

Der primäre Vorteil der Speicherzuordnung einer Datei erhöht die I / O-Leistung, besonders wenn sie auf großen Dateien verwendet wird. Bei kleinen Dateien können speicherzugeordnete Dateien zu einer Verschwendung von Leerzeichen führen, da Speicherkarten immer auf die Seitengröße ausgerichtet sind, die meistens 4 KiB beträgt. Daher wird eine 5-KiB-Datei 8 KiB zuweisen und somit werden 3 KiB verschwendet. Der Zugriff auf Speicher-abgebildete Dateien ist schneller als die Verwendung von direkten Lese- und Schreiboperationen aus zwei Gründen. Erstens ist ein Systemaufruf um Größenordnungen langsamer als eine einfache Änderung des lokalen Speichers eines Programms. Zweitens ist in den meisten Betriebssystemen der Speicherbereich, der tatsächlich abgebildet ist, der Cache des Kernels (Datei-Cache), was bedeutet, dass keine Kopien im Benutzerraum erstellt werden müssen.

Bestimmte benutzerdefinierte Speicher-abgebildete Datei-Operationen leisten mehr als ihrer physischen Datei Gegenstücke. Anwendungen können auf Daten in der Datei direkt und in-place zugreifen und diese aktualisieren, im Gegensatz zu der Suche nach dem Beginn der Datei oder dem Umschreiben der gesamten bearbeiteten Inhalte an einen temporären Ort. Da die Speicher-abgebildete Datei intern in Seiten behandelt wird, benötigt der lineare Dateizugriff den Datenträgerzugriff nur dann, wenn eine neue Seitengrenze überschritten wird und größere Abschnitte der Datei auf Festplatte in einer einzigen Operation.

Ein möglicher Vorteil von Speicher-abgebildeten Dateien ist ein "faules Laden", so dass kleine Mengen an RAM auch für eine sehr große Datei. Der Versuch, den gesamten Inhalt einer Datei zu laden, die signifikant größer ist als der verfügbare Speicherplatz, kann zu schweren Thrashs führen, da das Betriebssystem von der Festplatte in den Speicher liest und gleichzeitig die Seiten vom Speicher zurück auf die Festplatte schreibt. Memory-Mapping kann nicht nur die Seiten-Datei vollständig umgehen, aber das System muss nur die kleineren Seiten-Größe Abschnitte laden, wie die Daten bearbeitet werden, ähnlich wie die Nachfrage Paging-Schema für Programme verwendet.

POSIX Shared Memory Segments

Unix Domain Socket