Zertifikatsprogramm - Z202

Python 1 - Programmierung und Forensik

- Einführung in Python
- Forensische Analyse mit Python: Datenbanken und Anwendungen
- Forensische Analyse mit Python: Windows

Prof. Dr. Martin Rieger
Patrick Eisoldt, M.Eng.
David Schlichtenberger, M.Sc.

Open C³S
Open Competence Center for Cyber Security

Hochschule
Albstadt-Sigmaringen
Albstadt-Sigmaringen University

Python 1 – Programmieren im IT-Security-Umfeld

Studienbrief 1: Einführung in Python

Studienbrief 2: Forensische Analyse mit Python: Datenbanken und Anwendungen

Studienbrief 3: Forensische Analyse mit Python: Windows

Autoren:

Prof. Dr. Martin Rieger

Patrick Eisoldt, M.Eng.

David Schlichtenberger, M.Sc.

2. Auflage

Hochschule Albstadt-Sigmaringen

Institut für wissenschaftliche Weiterbildung
Open C^{3}S | Zertifikatsprogramm
Steinachstraße 11
72336 Balingen

2. Auflage (2017-01-25)

Um die Lesbarkeit zu vereinfachen, wird auf die zusätzliche Formulierung der weiblichen Form bei Personenbezeichnungen verzichtet. Wir weisen deshalb darauf hin, dass die Verwendung der männlichen Form explizit als geschlechtsunabhängig verstanden werden soll.

Inhaltsverzeichnis

Einleitung zu den Studienbriefen

I. Abkürzungen der Randsymbole und Farbkodierungen

Beispiel	B
Definition	D
Exkurs	E
Merksatz	M
Quelltext	Q
Übung	Ü

II. Zu den Autoren

Prof. Dr. Martin Rieger studierte Elektro- und Informationstechnik an der Technischen Universität München und schloss an derselben Hochschule die Promotion mit Auszeichnung ab. Ein Schwerpunkt seiner Forschungsarbeit lag in der Erstellung von Methoden und Werkzeugen zur Modellierung sowie Analyse und Optimierung elektrischer Schaltungen. Er war fünf Jahre Leiter des Labors für schnelle Analog-ICs in der IC-Entwicklungsabteilung des Forschungs- und Entwicklungszentrums der Firma Thomson Multimedia in Villingen. In der Zeit bei Thomson Multimedia war er Erfinder bzw. Miterfinder an 15 deutschen, 12 europäischen und 8 weltweiten Patenten.

Seit 1993 ist er als Professor an der Fakultät Engineering der Hochschule Albstadt-Sigmaringen auf den Gebieten Informatik und Informationstechnik tätig. Im Labor für Eingebettete Systeme und IT-Sicherheit betreibt er anwendungsnahe Forschung auf den Gebieten Embedded Systems und IT-Sicherheit. Er hatte über viele Jahre an der Hochschule Albstadt-Sigmaringen Positionen als Studiendekan, Prodekan, Prorektor und Rechenzentrumsleiter inne.

Prof. Dr. Rieger ist Initiator und Gründungs-Studiendekan des Master-Studiengangs Digitale Forensik, der in Kooperation mit der Friedrich-Alexander Universität Erlangen-Nürnberg und der Ludwig-Maximilians-Universität München betrieben wird.

Er leitet das vom BMBF geförderte Zertifikatsprogramm Open-C^{3}S, das 35 Hochschul-Zertifikatsmodule auf dem Gebiet Cybersicherheit anbietet und das kooperativ von der Hochschule Albstadt-Sigmaringen, der Friedrich-Alexander-Universität Erlangen-Nürnberg, der Freien Universität Berlin, und der Ludwig-Maximilians-Universität München, getragen wird.

Patrick Eisoldt, M. Eng. hat an der Hochschule Albstadt-Sigmaringen und der Glyndŵr University in Wales studiert. 2012 schloss er erfolgreich sein Masterstudium Systems Engineering ab. Im Rahmen seiner Master-Thesis konzipierte und realisierte er einen prototypischen Editor zur Projektierung von Prozessleitsystemen der Firma Siemens nach dem Ursache-Wirkung-Prinzip. Von November 2010 bis August 2011 unterstützte er das Institut für Wissenschaftliche Weiterbildung bei der Erstellung von Studienbriefen für den Studiengang Digitale Forensik.

Seit 2012 ist er für das Open Competence Center for Cyber Security als Modulentwickler tätig.

David Schlichtenberger studierte Medien- und Kommunikationsinformatik an der Hochschule Reutlingen. Nach seinem Masterstudium arbeitete er einige Jahre als Webentwickler und Kundenberater für Internetservices. Seit November 2014 ist er als akademischer Mitarbeiter an der Hochschule Albstadt-Sigmaringen am Institut für Wissenschaftliche Weiterbildung beschäftigt.

III. Modullehrziele

Ziel dieses Moduls ist es, Aufgabenstellungen aus dem Umfeld der IT-Sicherheit mit Hilfe von Python-Programmen schnell und effektiv lösen zu können. In diesem Modul lernen Sie die Programmiersprache Python anhand von praktischen Übungen kennen. Ziel dieses Moduls ist es nicht, Vorgehensmodelle zur Softwareentwicklung zu vermitteln, wie sie bei komplexer Software benötigt werden. Mit Python sollen Sie viel mehr in der Lage sein, kleinere überschaubare Programme zu schreiben, die schnell zu Ergebnissen führen.

Python liegt in zwei Versionen vor, die beide aktiv von Programmierern verwendet werden. Die Version 3 ist mit Python 2 nicht mehr kompatibel und ist die einzige Version, die aktiv weiterentwickelt wird. In diesem Modul wird weitestgehend die aktuelle Version von Python verwendet. In einigen Abschnitten wird aber auf die Vorgängerversion zurückgegriffen, da die verwendeten Module noch nicht von den Entwicklern portiert wurden.

Neben der Programmiersprache Python wird auch das Erstellen und Verwenden von Datenbanken grundlegend erklärt. Hierfür wird das Hilfsmodul SQLite verwendet, das ein wartungsfreies Datenbanksystem enthält und Teil der Python-Umgebung ist.

Der Studienbrief 1 beschäftigt sich mit den Grundlagen der Python-Programmierung, Studienbrief 2 mit Datenbanken. Studienbrief 2 schließt mit der Untersuchung von Anwendungsartefakten an den Beispielen Skype und Firefox ab.

Der Studienbrief 3 befasst sich mit dem Thema Informationsgewinnung unter forensischen Aspekten. Die vorgestellten Beispiele sollen dabei die universellen Einsatzmöglichkeiten von Python aufzeigen und zum Experimentieren einladen.

Auf dieses Modul wird mit einem Folgemodul „Python 2 – Programmieren im IT-Security-Umfeld“ aufgebaut, bei dem der Fokus auf Penetrationstests und Netzwerkforensik liegt.

Hinweise zur Typographie

Dieses Modul enthält zahlreiche Programmbeispiele, die stets in einer `Monospace Schriftart` gehalten sind. Zusätzlich wird zwischen Beispielen ohne gelbe Kästen und Quelltext in gelben Kästen unterschieden. Die Beispiele ohne gelbe Kästen sollen in der Python-Shell der IDLE-Umgebung (siehe Abs. 1.5) realisiert werden, Quelltexte in gelben Kästen in einer Skriptdatei (siehe Abs. 1.10). Zusätzlich können die Beispiele für die Shell an der sogenannten Prompt (>>>) erkannt werden. An einigen Stellen sind die Codezeilen zu lang, weshalb sie mit einem Backslash (\) umgebrochen wurden. Der Quellcode ist somit korrekt und kann wie abgebildet ausgeführt werden.

Warum Python?

Python ist eine interpretierte höhere Programmiersprache. Es handelt sich um eine leicht zu erlernende Programmiersprache, die sich durch ihren übersichtlichen und gut zu lesenden Quellcode auszeichnet. In der Python-Shell können Befehle und deren Wirkung direkt beobachtet werden. Dies erleichtert den Einstieg in das Programmieren und ermöglicht das „schnelle Ausprobieren“, ohne das Programm gleich vollständig implementieren zu müssen. Python wird oft als Skriptsprache verwendet, es unterstützt aber unterschiedliche, fundamentale Programmierstile (Programmierparadigma) und kann auch für größere Projekte eingesetzt werden. Einer der größten Stärken von Python ist die große Sammlung an standardisierten Programmkonstrukten (Funktionen innerhalb der Standardbibliothek), wodurch sich Python für viele Anwendungsbereiche eignet. Die meisten Funktionen sind zudem plattformunabhängig und sind somit auf unterschiedlichen Betriebssystemen lauffähig. Die Community von Python erweitert das Einsatzgebiet zusätzlich durch die unzähligen Open-Source-Projekte.

Ende 2013 erreichte Python Spitzenwerte bei der Softwarequalität und zeichnete sich durch einen äußerst hohen Reifegrad aus.[1]

Im Gegensatz zu den meisten verbreiteten Programmiersprachen verzichtet Python bei der Strukturierung auf eine Klammerung und begrenzt die Blöcke stattdessen durch gleiche Einrückung. Python unterstützt sowohl objektorientierte Programmierung als auch aspektorientierte oder funktionale Programmierung. Im Gegensatz zu Java, bei dem die Grunddatentypen keine Objekte sind, ist in Python alles ein Objekt, egal ob Klasse, Typ, Methode, Modul etc. Der Datentyp einer Variable wird dynamisch vergeben uns ist an das Objekt gebunden. Die verhältnismäßig wenigen Schlüsselwörter in Kombination mit der reduzierten Syntax tragen zu einem übersichtlichen Gesamtbild des Quellcodes bei. Ein Python-Programm ist oftmals kürzer als eine Implementierung in einer anderen Sprache wie C++ oder Java[2], was sich positiv auf die Entwicklungszeit auswirken kann.

Das Zen von Python

Ein offizieller Beitrag zu den Python-Verbesserungsvorschlägen (engl. Python Enhaucement Proposals, kurz PEPs) beinhaltet die Philosophie von Python und verdeutlicht anhand von 20 Aphorismen, von denen nur 19 niedergeschrieben wurden, welches Konzept sich hinter der Programmiersprache verbirgt.[3]

Diese Leitgedanken werden auch als „das Zen von Python“ bezeichnet:

Schön ist besser als hässlich.
Explizit ist besser als implizit.
Einfach ist besser als kompliziert.
Kompliziert ist besser als undurchschaubar.
Flach ist besser als verschachelt.
Spärlich ist besser als beschränkt.
Lesbarkeit zählt.
Spezialfälle sind nicht spezial genug, als dass sie die Regeln sprengen dürften.
Obwohl die praktische Anwendbarkeit die Reinheit übertrifft.
Fehler sollten nie schweigend verlaufen.
Außer man hat sie explizit zum Schweigen gebracht.
Im Angesicht der Mehrdeutigkeit, widersage der Versuchung zu raten.
Es sollten einen — und bevorzugt genau einen — offensichtlichen Weg geben, es zu tun.
Obwohl dieser Weg auf den ersten Blick nicht offensichtlich erscheinen mag, außer man ist Holländer.
Jetzt ist besser als nie.
Obwohl nie oft besser ist als JETZT SOFORT.
Wenn die Implementierung schwer zu erklären ist, ist es eine schlechte Idee.
Wenn die Implementierung einfach zu erklären ist, könnte es eine gute Idee sein.
Namensräume sind eine glänzende Idee — lasst uns mehr davon tun!

Obwohl die Formulierung scherzhaft ist und sich die Aufzählung als Easter Egg in der Python-Umgebung wiederfindet (`import this`), sind diese Aphorismen durchaus ernst gemeint und spiegeln sich bereits bei den einfachsten Programmen wieder. Hierfür werden in den Programmiersprachen Python, C++ und Java die Quelltexte für ein Hallo-Welt-Programm verglichen. Hierbei handelt es sich um ein Computerprogramm, dass auf einfachste Weise veranschaulicht, welche Bestandteile für ein lauffähiges Programm in einer Programmiersprache benötigt werden.

Quelltext 1: Hallo-Welt in Python

```
1 print("Hallo Welt!")
```

[1] `http://heise.de/-1948541` [Stand: 25.1.2017]
[2] `http://www.python.org/doc/essays/comparisons/` [Stand: 25.1.2017]
[3] `https://www.python.org/dev/peps/pep-0020/` [Stand: 25.1.2017]

Quelltext 2: Hallo-Welt in C++

```cpp
#include <iostream>

int main () {
  std::cout << "Hallo Welt!\n";
}
```

Quelltext 3: Hallo-Welt in Java

```java
public class HalloWelt
{
 public static void main(String[] args)
 {
  System.out.println("Hallo Welt!");
 }
}
```

Alle drei Programme würden beim Ausführen den gleichen Text an die Standardausgabe senden. Folglich würde der Text „Hallo Welt!“ auf dem Bildschirm erscheinen. Python benötigt hierfür, im Gegensatz zu den anderen Programmiersprachen, nur eine Zeile und verdeutlicht bereits damit eine Vielzahl der Leitgedanken. Das Programm ist auf das Wesentliche reduziert und enthält keine unnötigen und verwirrenden Bestandteile. Selbst ohne Programmierkenntnisse kann man verstehen, dass etwas ausgedruckt oder ausgegeben wird (`print`) und die Worte „Hallo Welt“ in irgend einem Verhältnis `()` zu `print` stehen.

Studienbrief 1 Einführung in Python

1.1 Lernergebnisse

Nach Abschluss dieses Studienbriefes sind Sie mit der Python-Shell vertraut und können mit Hilfe von Modulen aus der Standardbibliothek einfache Programme erstellen. Sie können die Begrifflichkeiten der objektorientierten Programmierung einordnen und können die wichtigsten Konzepte aufzählen. Mit dem Wissen über das Erstellen von eigenen Funktionen können Sie nicht nur auf die Funktionen von Modulen zurückgreifen, sondern darüber hinaus auch Ihr Programm besser strukturieren. Sie sind mit den Standarddatentypen in Python vertraut und können differenzieren, wann Sie welchen Datentyp verwenden sollen und wie sie eine Typumwandlung vornehmen. Zudem können Sie Skriptdateien erstellen und mithilfe von Kontrollstrukturen auch komplexere Programme erstellen. Hierbei unterstützt Sie auch das Wissen, wie ein Textmenü erstellt werden kann und Kommandozeilenparameter entgegengenommen werden. Innerhalb der Sortieralgorithmen sind Ihnen Bubblesort und Timsort bekannt, wobei Sie mit Bubblesort einen Sortieralgorithmus im Detail verstanden haben und mit Timsort einen effektiven Sortieralgorithmus kennen und zudem Aussagen zur Effizienz treffen können.

1.2 Advance Organizer

Skriptsprachen eignen sich besonders gut für schnelle Ad-hoc-Lösungen. Durch die Kenntnisse im Umgang mit Python ist es somit beispielsweise möglich, Aufgaben mit sich wiederholendem Ablauf zu automatisieren. Sie können zudem viele Programmierkonstrukte von Python auf andere Programmiersprachen übertragen, was das Verstehen und Erlernen von weiteren Programmiersprachen erheblich fördert.

1.3 Grundlagen

Das Modul ist sehr praxisorientiert und soll einen schnellen Einstieg in die Programmierung bieten. Dennoch werden in diesem Abschnitt einige Grundlagen vermittelt, die für das Verständnis des Moduls bzw. Programmieren im Allgemeinen unerlässlich sind.

1.3.1 Programmieren

Beim Programmieren geht es um das Formulieren eines Algorithmus' in einer Programmiersprache. Somit kann die Programmiersprache als Schnittstelle zwischen dem Menschen und einem Computer verstanden werden.

Definition 1.1: Algorithmus

Ein Algorithmus ist eine eindeutige, ausführbare Folge von Anweisungen endlicher Länge zur Lösung eines Problems. Ein Algorithmus besteht aus einem Deklarationsteil, der definiert was benötigt wird, und einem Anweisungsteil, der beschreibt wie das Problem gelöst wird.

D

Entscheidend für das Erlernen einer Programmiersprache ist das Denken in Strukturen und Algorithmen. Eine komplexe Aufgabenstellung muss in kleinere Teilprobleme aufgeteilt und diese wiederum durch Algorithmen gelöst werden.

Studienbrief 2 Forensische Analyse mit Python: Datenbanken und Anwendungen

2.1 Lernergebnisse

Sie können Informationen als Datensatz abbilden und diese logisch in Tabellen unterteilen. Sie sind imstande, einfache Datenbanken zu entwerfen und dabei die Regel der 1. Normalform zu beachten. Hierfür verwenden Sie das Ihnen bekannte ER-Modell nach der Chan-Notation. Da Sie mit der Datenbanksprache SQL vertraut sind, sind Sie nach kurzer Einarbeitungszeit in der Lage die Datenbank mit einer beliebigen Datenbank-Software umzusetzen. Durch diesen Studienbrief sind Sie im speziellen mit der Programmbibliothek „sqlite3“ vertraut. Mit der Untersuchung von Anwendungs-Artefakten in Python haben Sie den Aufbau der Skype-, Chrome- und Firefox-Datenbank verstanden und können hier gezielt Informationen extrahieren.

2.2 Advance Organizer

Bei der alltäglichen Arbeit am Rechner kommt man unbewusst vielfach mit Datenbanken in Berührung. Viele Anwendungen, wie Internet-Browser oder Chatprogramme, verwenden Datenbanken. Auch Webseiten, die mit einem Inhaltsverwaltungssystem (z. B. Joomla, TYPO3 oder WordPress) arbeiten, verwenden diese, um den eigentlichen Inhalt zu speichern. Diese Datenbanken können vor allem bei Anwendungen mit geringem Arbeitsaufwand ausgelesen werden und zum Beispiel nach konkreten Inhalten untersucht werden.

2.3 Datenbanksystem

Ein Datenbanksystem (DBS) ist ein System zum Speichern und Verwalten von meist großen Datenmengen. Ein DBS besteht aus einer Verwaltungssoftware, dem Datenbankmanagementsystem (DBMS) und den zu verwaltenden Daten, die als Datenbank bezeichnet werden.

Die Art und Weise, wie ein Datenbanksystem abgebildet wird, bezeichnet man als Datenbankmodell. Ein etablierter Standard ist hierbei die relationale Datenbank. Dabei handelt es sich um eine Sammlung von logisch zusammenhängenden Tabellen. Jede Zeile (Tupel) einer Tabelle ist ein Datensatz (record), wobei die Spalten die Eigenschaften (Attribute) der Tupel beschreiben.

Beispiel 2.1: Relation „Buch“

B

Ein Buch in einer Bibliothek soll durch einen Datensatz beschrieben werden. Der Datensatz besteht aus folgenden Attributen:

- Buch-ID
- Autor
- Titel
- Verlag
- Verlagsjahr

Um einen Datensatz **eindeutig** identifizieren zu können, wird ein (oder mehrere) Schlüssel (key) benötigt. Ein Schlüssel hat eine unveränderliche

Studienbrief 3 Forensische Analyse mit Python: Windows

3.1 Lernergebnisse

Sie können den Aufbau der Windows-Registry beschreiben und den Registry-Schlüssel zu gesuchten Informationen bestimmen und auswerten. Die Hauptschlüssel der Registry und deren Inhalt können Sie erläutern. Sie können die Aufgabe von SIDs, SAMs und GUIDs beschreiben und damit verbundene Informationen zuordnen. Bei einem Live System können Sie die Werte der Registry über das Python-Modul `winreg` extrahieren. Zudem ist es Ihnen möglich aus sichergestellten Hive-Dateien Informationen mit dem Modul `python-registry` zu gewinnen. Sie sind mit Python in der Lage gelöschte Dateien aus dem Papierkorb wiederherzustellen und gespeicherte WLAN-Kennwörter zu entschlüsseln. Ihr Wissen über die Windows-Forensik umfasst zudem das Parsen von Metadaten bei PDFs, Bilddateien, Word-Dokumenten und Torrent-Dateien sowie dateiformatsunabhängigen Metadaten.

3.2 Advance Organizer

Die Registry wird als „Goldgrube“ für forensisch interessante Spuren in Windows-Clients bezeichnet, weil sie von vielen Nutzeraktivitäten umfassende Spuren enthält. Dieser Studienbrief soll Ihnen als Leitfaden für die forensische Nutzung der Windows-Registry dienen und Ihnen einen Einblick in die Struktur und deren Besonderheiten verschaffen.

3.3 Windows Registry

Die Registry gibt es seit der Betriebssystemversion Windows 95. Sie ist als zentrale, hierarchisch organisierte Datenbank angelegt. Ihre binäre Form ermöglicht einen schnelle konvertierungsfreie Verarbeitung, die durch die Transaktionsüberstützung ganz oder gar nicht durchgeführt werden. Die Registry ist eine unverzichtbare Quelle für Forensiker, denn sie enthält wichtige Informationen über die Systemkonfigurationen, die Benutzerprofile und -aktivitäten, die installierten Programme und deren Ausführungen sowie über Hard- und Software-Komponenten des Rechnersystems.

3.3.1 Einführung

Die Registry bildet ein baumartiges, hierarchisch aufgebautes System, ähnlich wie Dateien und Ordner. In der Registry werden die Informationsbehälter als „Schlüssel“ (HKEY) bezeichnet. Diese sind den Dateiordnern ähnlich. Schlüssel können Unterschlüssel (KEY) haben, genau wie Ordner Unterordner haben können (vgl. Abb. 3.1). Die Daten, die in einem Schlüssel enthalten sind, werden als „Wert“ bezeichnet und sind vergleichbar zu Dateinamen. Die eigentlichen Daten können unterschiedliche Formate (z. B. Zeile, Zahl oder Zahlenfolge) haben.

selbst erst am 11. März 2015 auf dem Datenträger kopiert wurde. Seit Erstellung folgten keine Änderungen an der Datei. Das Änderungsdatum wurde von der Datei selbst übernommen.

K

Kontrollaufgabe 3.17

Gegeben sei ein Computer mit Windows 7 und zwei Datenträger. Diese sind jeweils mit dem NTFS-Dateisystem formatiert.

Welche der untenstehenden Ereignisse wirken sich auf welche Zeitstempel aus?

- Eine Datei wird von Datenträger 1 auf Datenträger 2 kopiert.
- Eine Datei wird von Datenträger 1 auf Datenträger 2 verschoben.
- Eine Textdatei mit dem Inhalt „Hallo“ wird mit dem selben Inhalt überschrieben.
- Eine ausführbare Datei wird mittels Doppelklick ausgeführt.

3.8 Zusammenfassung

Die Windows-Registry ist eine hierarchische Datenbank und dient als zentrale Instanz aller Konfigurationen. In der Registry sind zum Beispiel die physikalischen Adressen hinterlegt, mit denen sich der Rechner über WLAN verbunden hat. Als weiteres Beispiel für die Einträge in der Registry können die Windows-Benutzernamen genannt werden. Mit dem Modul `winreg` lassen sich diese Informationen auslesen.

Metadaten sind optionale Informationen, die für den Anwender nicht direkt sichtbar sind. Die Metadaten eines PDFs enthalten zum Beispiel Informationen über den Autor. Exif ist ein Standard für Metadaten von JPEG- und TIFF-Bildern. Der Umfang der Exif-Daten variiert stark und hat abhängig von der verwendeten Hardware und den Einstellungen einen hohen Informationsgehalt. So verfügen die meisten Smartphones über die Möglichkeit, bei Bildern die GPS-Koordinaten als Metadaten zu hinterlegen. Die Metadaten können über externe Module der Python-Community ausgelesen werden.

Beautiful Soup ermöglicht das Parsen von HTML- und XML-Dokumenten. In diesem Studienbrief wurde gezeigt, wie alle Bilder einer Seite mit *Beautiful Soup* herausgefiltert und heruntergeladen werden können und wie diese Bilder anschließend auf GPS-Koordinaten untersucht werden können.

3.9 Übungen

Ü

Übung 3.1

Xpdf[29] stellt unter anderem das Programm `pdftotext.exe` bereit, das eine PDF-Datei in eine Textdatei umwandeln kann. Es gibt auch Python-Module, die das Umwandeln von PDFs in Textdateien beherrschen, diese sind aber laut Entwickler um das 20-fache langsamer[30]. Erstellen Sie ein Python-Skript, das unter Verwendung von `pdftotext` eine PDF-Datei konvertiert.

Ü

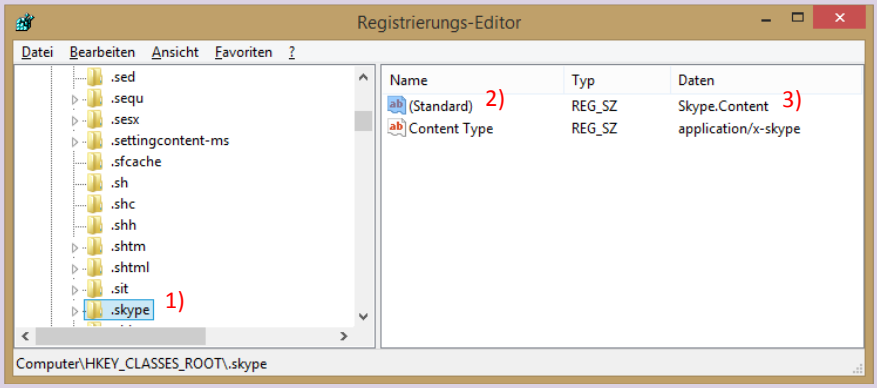
Übung 3.2

Ergänzen Sie das Skript aus Übung 3.1. Das Skript soll ein Durchsuchen der erstellten TXT-Datei nach einem bestimmten Suchbegriff ermöglichen. Versuchen Sie eine Lösung zu realisieren, die die Zeilen ausgibt, in der sich der Suchbegriff befindet. Alternativ genügt eine Meldung über die Anzahl der gefundenen Treffer. Der Name der zu konvertierenden PDF-Datei sowie der Suchbegriff sollen als Parameter an das Skript übergeben werden.

Ü

Übung 3.3: Registry

Erstellen sie ein Python-Skript, dass vollständig durch die Baumstruktur einer sichergestellten Hive-Datei (Post-Mortem-Analyse) iteriert und dabei die (Unter-)Schlüssel (1) sowie die Bezeichner (2) und die Daten (3) der Schlüsselwerte nach einem String durchsucht, der dem Skript als Argument übergeben wird.



Ü

Übung 3.4: WLAN-Schlüssel

Erstellen Sie ein Python-Skript, dass alle Dateien in den Verzeichnissen C:\ProgramData\Microsoft\Wlansvc\Profiles\Interfaces\[GUID] analysiert und die SSIDs sowie Kennwörter in Klartext auf der Konsole ausgibt.

[29] `http://www.foolabs.com/xpdf/download.html` [Stand: 25.1.2017]
[30] `http://www.unixuser.org/~euske/python/pdfminer/` [Stand: 25.1.2017]

Verzeichnisse

I. Abbildungen

II. Beispiele

III. Definitionen

IV. Exkurse

V. Tabellen

VI. Literatur

Grant Allen and Mike Owens. *The Definitive Guide to SQLite*. Apress, New York, 2nd edition, 2010. ISBN 978-1-430-23225-4.

Johannes Ernesti and Peter Kaiser. *Python 3 - Das umfassende Handbuch*. Galileo Press, 3. Auflage, Bonn, 2012. ISBN 978-3-836-21925-9.

F. Hajji. *Das Python-Praxisbuch: Der große Profi-Leitfaden für Programmierer*. Open source library. Pearson Deutschland, 2008. ISBN 9783827325433. URL `https://books.google.de/books?id=vfzazxfQScYC`.

D. Richard Hipp. Syntax diagrams for sqlite. Website, 2015. `http://www.sqlite.org/syntaxdiagrams.html#select-stmt`.

J. Honeycutt. *Microsoft® Windows® Registry Guide*. Microsoft Press, 2009. ISBN 9780735637351.

Jay A. Kreibich. *Using SQLite*. O'Reilly Media, Inc., Sebastopol, CA, 2010. ISBN 978-0-596-52118-9.

TJ O'Connor. *Violent Python - A Cookbook for Hackers, Forensic Analysts, Penetration Testers, and Security Engineers*. Newnes, London, 1st edition, 2012. ISBN 978-1-597-49957-6.

A.S. Tanenbaum. *Moderne Betriebssysteme*. Pearson Studium - IT. Pearson Deutschland, 2009. ISBN 9783827373427.

tutorialspoint. Sqlite - SQL Database Engine. Website, 2010. `http://www.tutorialspoint.com/sqlite/sqlite_using_joins.htm`.

Michael Unterstein and Günter Matthiessen. *Relationale Datenbanken und SQL in Theorie und Praxis*. Springer DE, 5. Auflage, Berlin, 2012. ISBN 978-3-642-28986-6.

Michael Weigend. *Objektorientierte Programmierung mit Python 3 - Einstieg, Praxis, professionelle Anwendung*. Hüthig Jehle Rehm, 4. Auflage, München, 2010. ISBN 978-3-826-61750-8.

Wikipedia. Office open xml. Website, 2015a. `http://de.wikipedia.org/wiki/Office_Open_XML`.

Wikipedia. Iptc-iim-standard. Website, 2015b. `http://de.wikipedia.org/wiki/IPTC-IIM-Standard`.

Anhang

A. Schlüsselwörter

```
False      class      finally    is         return
None       continue   for        lambda     try
True       def        from       nonlocal   while
and        del        global     not        with
as         elif       if         or         yield
assert     else       import     pass
break      except     in         raise
```

Stichwörter

Open C³S
Open Competence Center for Cyber Security

Fort- und Weiterbildung

Neue Bedrohungszenarien stellen Sicherheitsexperten und IT-Verantwortliche in Unternehmen und einschlägigen Behörden vor immer größere Herausforderungen. Neue Technologien und Anwendungen erfordern zusätzliches Know-how und personelle Ressourcen.

Zur Erhöhung des Fachkräftepools und um neues Forschungswissen schnell in die Praxis zu integrieren, haben sich die im Bereich lehrenden und forschenden Verbundpartner zum Ziel gesetzt, ein hochschuloffenes transdisziplinäres Weiterbildungsprogramm im Sektor Cyber Security zu entwickeln. Auf der Grundlage kooperativer Strukturen werden wissenschaftliche Weiterbildungsmodule im Verbund zu hochschulübergreifenden Modulpaketen und abschlussorientierten Ausbildungslinien konzipiert und im laufenden Studienbetrieb empirisch getestet.

Die Initiative soll High Potentials mit und ohne formale Hochschulzugangsberechtigung über innovative Weiterbildungsangebote (vom Zertifikat bis zum Masterprogramm) zu Sicherheitsexperten aus- und fortbilden. Hierzu werden innovative sektorale Lösungen zur Optimierung der Durchlässigkeit von berulicher und hoch- schulischer Bildung entwickelt und für eine erfolgreiche Implementierung vorbereitet. Unter prominenter Beteiligung einschlägiger Verbände, der Industrie sowie Sicherheits und Ermittlungsbehörden verfolgt die Initiative das Ziel, im deutschsprachigen Raum eine Generation von Fachkräften wissenschaftlich aus- und weiterzubilden, die unser Internet schützen kann.

Open Competence Center for Cyber Security

Open C³S ist aus dem Verbundvorhabens Open Competence Center for Cyber Security entstanden. Das Gesamtziel des Programms war die Entwicklung eines hochschuloffenen transdisziplinären Programms wissenschaftlicher Weiterbildung im Sektor Cyber Security. Das Bundesministerium für Bildung und Forschung (BMBF) fördert das Großprojekt im Rahmen des Wettbewerbs „Aufstieg durch Bildung: offene Hochschulen“, der aus BMBF-Mitteln und dem Europäischen Sozialfonds finanziert wird.

Neun in Forschung und Lehre renommierte Hochschulen und Universitäten aus dem gesamten Bundesgebiet haben sich zum Ziel gesetzt, Online-Studiengänge auf dem Gebiet der Cybersicherheit zu entwickeln. Dieses Konzept soll den Studierenden ermöglichen, sich berufsbegleitend auf hohem Niveau wissenschaftliche Qualifikationen anzueignen und akademische Abschlüsse zu erlangen. Beruflich erworbene Kompetenzen können eingebracht werden. Die Bezeichnung „Open“ steht auch für die Öffnung des Zugangs zu akademischer Bildung ohne klassischen Hochschulzugang.

Mission der Initiative ist es, dringend benötigte Sicherheitsexperten aus- und fortzubilden, um mit einer sicheren IT-Infrastruktur die Informationsgesellschaft in Deutschland und darüber hinaus zu stärken.

Umsetzungsnahes Wissen ist ein wesentlicher Schlüssel um der wachsenden digitalen Bedrohung zu begegnen. Solange wir nicht in der Lage sind, Systeme hinreichend zu härten, Netzwerke sicher zu designen und Software sicher zu entwickeln, bleiben wir anfällig für kriminelle Aktivitäten. Unser Ziel ist es, die Mitarbeiter von heute zu Sicherheitsexperten und Führungskräften von morgen auszubilden und dafür zu sorgen, dass sich die Zahl und die Fertigkeiten dieser Experten nachhaltig erhöht.

Weitere Informationen zu Open C³S: https://open-c3s.de

Z202 Python 1 - Programmierung und Forensik

Ziel dieses Moduls ist es, Aufgabenstellungen aus dem Umfeld der IT-Sicherheit mit Hilfe von Python-Programmen schnell und effektiv lösen zu können. In diesem Modul lernen Sie die Programmiersprache Python anhand von praktischen Übungen kennen. Ziel dieses Moduls ist es nicht, Vorgehensmodelle zur Softwareentwicklung zu vermitteln, wie sie bei komplexer Software benötigt werden. Mit Python sollen Sie viel mehr in der Lage sein, kleinere überschaubare Programme zu schreiben, die schnell zu Ergebnissen

führen.

Python liegt in zwei Versionen vor, die beide aktiv von Programmierern verwendet werden. Die Version 3 ist mit Python 2 nicht mehr kompatibel und ist die einzige Version, die aktiv weiterentwickelt wird. In diesem Modul wird weitestgehend die aktuelle Version von Python verwendet. In einigen Abschnitten wird aber auf die Vorgängerversion zurückgegriffen, da die verwendeten Module noch nicht von den Entwicklern portiert wurden.

Neben der Programmiersprache Python wird auch das Erstellen und Verwenden von Datenbanken grundlegend erklärt. Hierfür wird das Hilfsmodul SQLite verwendet, das ein wartungsfreies Datenbanksystem enthält und Teil der Python-Umgebung ist.

Der Studienbrief 1 beschäftigt sich mit den Grundlagen der Python-Programmierung, Studienbrief 2 mit Datenbanken. Studienbrief 2 schließt mit der Untersuchung von Anwendungsartefakten an den Beispielen Skype und Firefox ab.

Der Studienbrief 3 befasst sich mit dem Thema Informationsgewinnung unter forensischen Aspekten. Die vorgestellten Beispiele sollen dabei die universellen Einsatzmöglichkeiten von Python aufzeigen und zum Experimentieren einladen.

Auf dieses Modul wird mit einem Folgemodul „Python 2 – Programmieren im IT-Security-Umfeld“ aufgebaut, bei dem der Fokus auf Penetrationstests und Netzwerkforensik liegt.

Zertifikatsprogramm

Die Zertifikatsmodule auf wissenschaftlichem Niveau und mit hohem Praxisbezug bilden ein passgenaues Angebot an Qualifikation und Spezialisierung in der nebenberuflichen Weiterbildung. Damit können einzelne Module nebenberuflich studiert werden. Durch die Vergabe von ECTS-Punkten können sie auf ein Studium angerechnet werden.

https://zertifikatsprogramm.de