# Teilnehmer/innen des Teams:

|  |  |
| --- | --- |
| Klasse:  AP18c | Team:  Schuhmacher, Anderfuhren |

# Anforderungsdefinition (Meilenstein A)

|  |  |
| --- | --- |
| „Netter Virus“ | |
| **Fachlicher Inhalt:**  (Allgemeine Beschreibung) | Nutzen: Das Programm ist ein Virus, aber kein wirklich bösartiger. Daher hat er keinen Nutzen, ausser wenn man den Computer nicht mag. In diesem Fall kann man mit dem Virus den Speicher mit Virus-Dateien füllen.**Automation:**  Das Programm lädt eine Kopie der neuesten Version von sich selbst von einem Server herunter, speichert sie, und startet dann diese. Dafür ist keine Benutzereingabe nötig.  **Details:**   * Konfiguration: Automatisch. (Durch die Version auf dem Server.) * Integration: Virus. (Keine Benutzerinteraktion erforderlich.) * Administration: Durch die Version auf dem Server. * Sicherheitsaspekte: Unsicher.   **Skizze**:    **Erkenntnisse aus der Machbarkeitsabklärung:**  Im Prinzip funktioniert es wie diese Bash-Skripte, die sich selbst zweimal aufrufen und den Computer dadurch zum Absturz bringen. |
| **MUSS**  **Kriterien:**  (Konkrete Features, die umzusetzen sind) | **Folgende Features sollen implementiert werden, um einen produktiven Ablauf sicherzustellen:**   * Herunterladen der neuesten Version des Virus (FTP) * Speichern der neuesten Version auf dem Filesystem * Wartezeit (erlaubt dem Benutzer/Opfer/Entwickler, den Virus zu stoppen, ohne dass sofort der ganze Computer unbrauchbar wird) * Die neue Version des Virus starten |
| **KANN**  **Kriterien:**  (Konkrete Features, die optional sind) | **Folgende Features können zusätzlich implementiert werden: (Varianten, Kreativität)**   * Dem Benutzer etwas Nettes oder Nützliches sagen (erledigt) * Benutzer kann Wartezeit angeben (abgelehnt, da anfällig auf Benutzerfehler) * Log des Angriffs erstellen (erledigt) * Logfiles zippen (abgelehnt, da Log ein TXT File ist, wo zippen nicht viel bringt) * Mail an Ersteller des Virus, dass ein Angriff erfolgt ist (abgelehnt, da dies zu viele Mails wären) * Logfiles oder Zips ans Mail anhängen (abgelehnt, da kein Mail) |

## Planung Meilensteine (LB1 / LB2)

|  |  |  |  |
| --- | --- | --- | --- |
| *MS* | *Tätigkeit / Abgabe* | *Soll-Datum* | *Ist-Datum* |
| A | ProjektstartTeam Bildung, Kollaborationsplattform, GitHub Repos \*, LehrerzugangWahl / Ausarbeitung der Anforderungsdefinition Abnahme Anforderungsdefinition durch Lehrperson | 21.05.2021 |  |
| B | Teamaufgabe 1:Abgabe: Lösungsdesign  (Funktionsmodell / GUI / PAP / Storyboard) | 28.05.2021 |  |
| C | Einzelaufgabe 2 (LB1):Abgabe Programmcode und DokumentationFachgespräch Projektabnahme | 11.06.2021 |  |

\*) Öffentliche GitHub-URLs im Ablageordner auf dem BSCW ablegen! (pro Team)

Namenskonvention URL: **M122\_Klasse\_Thema\_Name\_Name**

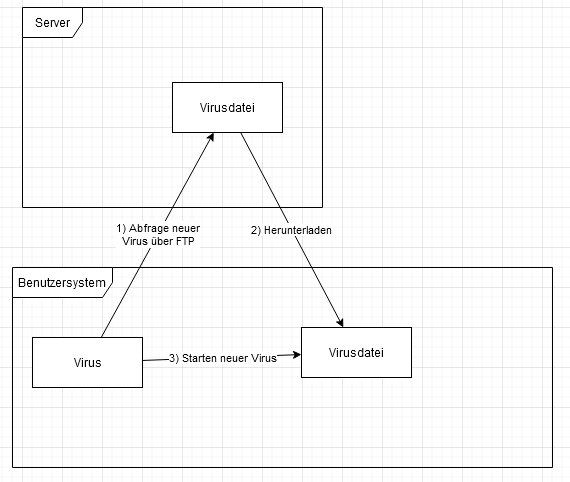
# Lösungsdesign (Meilenstein B: Teamaufgabe 1)

Anhand der Analyse wurde folgendes Lösungsdesign entworfen:

## Schematische Darstellung der Funktionalität, sog. Funktionsmodell

Im Folgenden ist die erwartete Funktionalität dargestellt und erklärt:

...



Server: Server, auf dem der Quellcode des Virus liegt.

Benutzersystem: System, auf dem das Virus läuft.

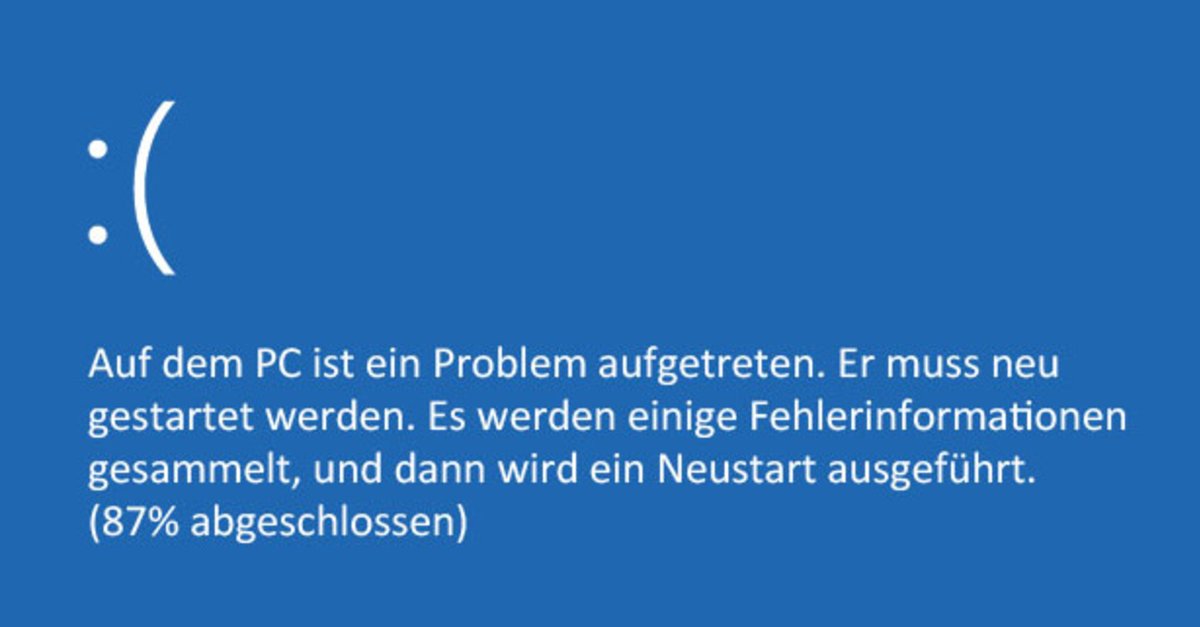
Virus: Das aktuell laufende Virus-Programm.

Virusdatei: Datei mit dem Quellcode des Virus.

## Graphische Benutzer Schnittstelle (GUI) zur Konfiguration des Ablaufs

Das zu erwartende GUI ist hier dargestellt:

Wenn der Virus nicht gestoppt wird, ist nach einiger Zeit folgende Anzeige vom Betriebssystem zu erwarten:



## Ablauf der Automation

Aus Benutzersicht ist folgender Ablauf des Programms zu erwarten:

Aus Benutzersicht ist nichts zu erwarten, ausser einem Computer-Absturz und einem Volllaufen der Festplatte.

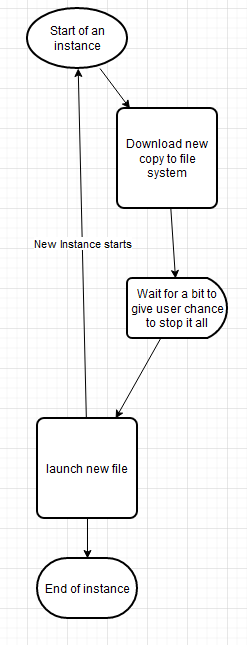
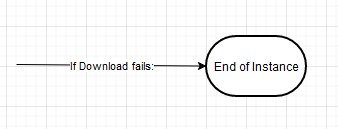
Das Kann-Feature «Nettes oder Nützliches sagen» gibt dem Benutzer eine Nachricht.

Aus Administratorsicht ist folgender Ablauf des Programms zu erwarten:

Der Administrator nimmt keinen Einfluss auf das Geschehen, und wird davon auch nicht beeinflusst.

Das Kann-Feature «Mail an Erschaffer des Virus» informiert ihn über den Verlauf des Angriffs.

Abflussdiagramm:



# Systemdokumentation (Meilenstein C: individuelle Aufgabe 3)

Die erstellten Projekt-WPS-Scripts sind hier abgelegt und für Entwickler dokumentiert:

Öffentliche GitHub-URLs im Ablageordner auf dem BSCW ablegen! (pro Team)

Namenskonvention URL: **M122\_Klasse\_Thema\_Name\_Name**

* Ein* ***Branch*** *und separater Doku-Ordner pro Teammitglied erstellen*

## Umfang / Abgrenzung / Änderungen gegenüber Design

Aufgrund unten beschriebener Umstände sind Anpassungen des ursprünglichen Lösungsdesigns gemacht worden:

...

Anpassung 1: mehrere Kann-Features abgelehnt:

- Benutzer kann Wartezeit angeben: Abgelehnt, da sehr wahrscheinlich irgendwann ein Benutzer eine Wartezeit von null oder quasi null angeben wird, was den Virus zu gefährlich werden lässt.

- Mail an Ersteller des Virus: Abgelehnt, da, wenn der Virus auf mehreren Geräten ausgeführt wird, sehr schnell sehr viele Mails zusammenkommen werden. Dies würde zu Problemen im empfangenden Mailkonto führen.

- Zip der Logs ans Mail anhängen: Abgelehnt, da kein Mail.

- Logs Zippen: Abgelehnt, da es nur ein TXT-File ist. Dieses zu zippen, und dann nicht zu verschicken, macht keinen Sinn. Deshalb, und um Ressourcen zu sparen, wird das Logfile nicht gezippt.

Anpassung 2: Ablauf gegenüber Diagramm verändert: Zuerst warten, dann herunterladen. Begründung: Um den Schaden zu minimieren, wird das neue File erst heruntergeladen und gespeichert, nachdem der Virus gewartet hat. Dies ist daher eine Verbesserung gegenüber dem Plan, da somit, wenn der Virus abgebrochen wird, kein Download mehr durchgeführt werden muss, was die Last auf dem Server und dem Endgerät reduziert, und zusätzlich eine Erleichterung ist, wenn sich das Endgerät in einem Netz mit beschränkter Datenübertragung befindet.

## Funktionalität der Implementation.

Zusätzlich zu der Inline-Dokumentation sind hier folgende Funktionen / Eventhandler detailliert beschrieben:

Das meiste am Code, was nicht direkt inline kommentiert ist, gehört zur Funktionalität, die den Namen für die neue Datei ermittelt.

# Betriebsdokumentation (Meilenstein C: individuelle Aufgabe 3)

Für Administrator und Benutzer wird folgende Anleitung ausgeliefert ...

## Installationsanleitung für Administratoren

Das Programm ist folgendermassen zu installieren und konfiguriert ...  
  
Keine Konfiguration möglich.

Installation durch download an einen beliebigen Ort auf einem System, auf dem python installiert ist. Es sind keine Zusatzmodule nötig. Ich verwende Python 3.9.5, es kann aber auch sein, dass es auch mit einer anderen Python-Version funktioniert.

Warnung: Dateinamen nicht ändern, da es sonst zu einem Absturz des Programms führen kann. Ein Prozess in der Datei ist abhängig vom Dateinamen, und führt zu einem kritischen Fehler, wenn die Länge des Dateinamens nicht mehr stimmt.

## Bedienungsanleitung für Benutzer

Das Programm ist folgendermassen zu bedienen ...  
  
Start des Programms: Befehl «python» gefolgt vom Dateinamen im Terminal eingeben.

Stopp des Programms: Mehrere Varianten möglich:

- Unterbruch der Internetverbindung

- Schliessen des aktiven Terminals (während der Wartezeit)

- Abbruch des Terminalprogramms durch ctrl-c (während der Wartezeit)

- Stopp des Computers

Der Ersteller des Virus kann sämtliche laufenden Instanzen des Virus anhalten lassen, indem das File, das jeweils heruntergeladen und als neue Instanz gestartet wird, so verändert wird, dass es das nächste heruntergeladene File nicht mehr startet.