Zweck

i-doit ist ein webbasiertes Tool für das Inventory-Management. Mit ihm sollen Ressourcen der Universität Freiburg erfasst werden.

Den Ressourcen lassen sich Ansprechpartner zuordnen. Damit hilft es auch, im Bedarfsfall Process-Owner und Business-Owner zu identifizieren. Es erfüllt damit auch Grundfunktionen einer Ansprechpartner-Datenbank.

An i-doit lässt sich OTRS als Ticketsystem andocken, so dass sich Verfälle systematisch und personenübergreifend bearbeitet werden können. i-doit dient so als Rückgrat einer Vorfallsbearbeitung.

Die in i-doit erfassten Informationen sind so vormodelliert, dass eine Dokumentation für den Grundschutz nach BSI vorstrukturiert ist.

High-Level ToDos

- Mandantenfähigkeit vorbereiten
- Use-Cases| entwickeln
    - Maschinensäle
    - Pools
    - Netzwerk-Bereiche
- automatisierte Auswertungen

Organisation

Koordination geplant durch Dominik Bernhard, interimsmäßig durch JL.

Protokolle von Besprechungen

Aufgabenbereiche

Beschreibung von IT-Equipment

Vermutlich will man verschiedene IT-Geräte (alles, was in unserem Netz eine IP-Adresse bekommt und nicht direkt Netzwerk-Kern-Equipment ist) entsprechend inventarisieren wollen. Dabei soll eine Unterscheidung nach Art des Gerätes:

- ListenpunktClient (Desktop-PC, Laptop, Endbenutzergerät) - dieses wird im Moment im Projekt Boot-Auswahl-Server bearbeitet

- Hardware-Server
- Virtueller Server
- SIP-Telefone
- Drucker
- …

erfolgen. Damit kann auf konfigurationsmäßige Besonderheiten und die jeweiligen Bedürfnisse für die Betreiber besser eingegangen werden.

Dynamische Daten, wie S.M.A.R.T. Werte oder Speicherfehler etc. sollten im Monitoring vorgehalten werden und nicht im I-doIT.

Client

Wegen der Umsetzung eines Prototypens für eine Erstregistrierung im Univ. PC-Pool-Konzept, soll erstmal für klassische PC's, die über den Campus verteilt in den Pools betrieben werden, gesammelt werden, welche Daten relevant sein könnten:

Hardware

Es wäre mühsam hier Nutzer das in Freitextfelder eintragen zu lassen, was auch am Ende auch nicht wirklich gut auswertbar wäre. Deshalb schauen, was sich automatisch auslesen lässt?

- Typ der Maschine, Bezeichnung (potenziell: Manufacturer, Model)
- CPU: Wie angeben, wie Multi-CPU (müsste I-doIT auch schon was haben)
- RAM
- Festplatte/SSD (wie mehrere?)
- Typ Netzwerkkarte, MAC-Adresse, Typ Netzwerkkabel (SFP, TP, …)
- eigene Seriennummer / UUID (potenziell: Serial Number / ProductID)
- UEFI/BIOS
    - Secure Boot on/off
- TPM (Public Key)

Software/Zertifikate

- Landeslizenzen (z.B. Microsoft, Matlab usw.)
- Campuslizenzen
- Sammellizenzen
- Hinterlegung der Verträge der o.g. Lizenzgruppen (soweit bekannt)
- Einzellizenzen (soweit bekannt)
- Inwieweit kann eine Verknüpfung bestehender Lizenzen zu Rechnern realisiert werden?
- Zertifikate für TLS
- Zertifikat für TPM (Trusted Platform Module, da können weitere Sicherheitsfeatures dran hängen, da will man den Public-Teil irgendwo ablegen)

Netzwerk

Was lässt sich automatisieren (s.o. was kann ausgelesen werden, welche Infos muss der Nutzer angeben und welche kann man zur Vorauswahl bereits zur Verfügung stellen)?

- LAN / WLAN
- IPv4, v6 (Adressen)
- FQDN
- Firewall, Sicherheitslevel
- Netzwerksegment (Sicherheitslevel, Segmentierung)
- Anschluss (-hierarchie, Dose, Switch, Router, …)

Lokalisierung

- Zuordnung Raum
- Zuordnung Netzwerkdose

Sonstige Metadaten

- Eigentümer
- Kostenstelle
- Seriennummer
- UUID (System-unique), GUID (OS-System-unique)
- Beschaffungsdatum
- …

Artikel zum tag: pc-arbeitsplaetze, Hardware - Artikelübersicht, Software - Artikelübersicht

Server

Vermutlich Wiederholung vieler Punkte oben

Erweiterungen

- vermutlich mehrere Netzwerkkarten
- IPMI
- VLANs
- Höheneinheiten

Instanzen in der Cloud

- Zugewiesene Ressourcen: CPU, RAM, Storage
- IP-Adresse (IPv4, v6)
- Eigentümer
- Kostenstelle
- Ablaufdatum

Instanzen im ESX

- Zugewiesene Ressourcen: CPU, RAM, Storage

Client-Programmierung

Ansprechpartner für Client-Programmierung ist Jannik Schönartz, Mail thejannik@yahoo.de.

Beispiel eines Clients in i-Doit mit der Zuordnung

Test_Client im i-Doit

Bezeichnung

```
Test_Client     * Client -> General -> Title
Pool PC         * Client -> General -> Purpose
```

Typ der Maschine, Seriennummer, Service Tag

```
Fujitsu         * Client -> Model -> Manufacturer
P910            * Client -> Model -> Model
SPC0CU1AR       * Client -> Model -> Service Tag
PF-009R97L      * Client -> Model -> Serial number
```

CPU

```
3470            * Client -> CPU -> Title
Intel           * Client -> CPU -> Manufacturer
Core i5         * Client -> CPU -> Type
4           * Client -> CPU -> CPU Cores
3.2 Ghz         * Client <nspages -h1 -subns -exclude:start>-> CPU -> CPU
frequency
1           * Client -> CPU -> Quantity to create
```

RAM

```
2           * Client -> Memory -> Quantity
Kingston        * Client -> Memory -> Manufacturer
```

```
DDR3            * Client -> Memory -> Type
8 GB            * Client -> Memory -> Capacity
```

Festplatte/SSD

```
System          * Client -> Direct Attached Storage -> Device -> Title
SSD         * Client -> Direct Attached Storage -> Device -> Type
1           * Client -> Direct Attached Storage -> Device -> Devices to
create
Samsung         * Client -> Direct Attached Storage -> Device ->
Manufacturer
840         * Client -> Direct Attached Storage -> Device -> Model
240 GB          * Client -> Direct Attached Storage -> Device -> Capacity
SATA lll        * Client -> Direct Attached Storage -> Device -> Connection
```

```
Daten           * Client -> Direct Attached Storage -> Device -> Title
Hard Disk       * Client -> Direct Attached Storage -> Device -> Type
1           * Client -> Direct Attached Storage -> Device -> Devices to
create
Western Digital     * Client -> Direct Attached Storage -> Device ->
Manufacturer
WD30EURX        * Client -> Direct Attached Storage -> Device -> Model
3 TB            * Client -> Direct Attached Storage -> Device -> Capacity
SATA lll        * Client -> Direct Attached Storage -> Device -> Connection
```

Typ Netzwerkkarte, MAC-Adresse, IP Adresse

```
nic0            * Client -> Network -> Interface -> Title
Intel           * Client -> Network -> Interface -> Manufaturer
82579LM         * Client -> Network -> Interface -> Model
```

```
eth0            * Client -> Network -> Port -> Title
nic0            * Client -> Network -> Port -> Connected Interface
Ethernet        * Client -> Network -> Port -> Type
Standard        * Client -> Network -> Port -> Mode
Automatic       * Client -> Network -> Port -> Negotiation
Full            * Client -> Network -> Port -> Duplex
1 Gbit/s        * Client -> Network -> Port -> Speed
12:34:56:78:90:ab   * Client -> Network -> Port -> MAC-address
```

```
DHCP            * Client -> Host address -> Address allocation
10.1.2.3        * Client -> Host address -> IPv4 address
255.255.255.0       * Client -> Host address -> Netmask
Test-client.rz.privat   * Client -> Host address -> Hostname (FQDN)
eth0            * Client -> Host address -> Assigned port
```

Betriebsystem

```
Debian 9 x64        * Client -> Operating system -> Operating system
```

Standort

```
Institutsgebiet > Universitätsrechenzentrum > 1. Kellergeschoß > R -113 (UG)
* Client -> Location -> Location
```

Inventarnummer, Kostenstelle, etc

```
RZ099073        * Client -> Accounting -> Inventory number
4000203001      * Client -> Accounting -> Account (Kostenstelle)
02.08.2018      * Client -> Accounting -> Order date
03.08.2018      * Client -> Accounting -> Delivery date
03.08.2021      * Client -> Accounting -> Date of invoice
399,99€         * Client -> Accounting -> Investment costs
```

Beispiel eines Gebäudes in i-Doit mit der Zuordnung

```
Universitätsrechenzentrum   * Building -> General -> Title (HIS gebaeude)
```

```
Institutsgebiet         * Building -> Location -> Location (HIS campus)
784.832.046.087.579     * Building -> Location -> Longitude (HIS
geb_geogr_laenge)
4.800.364.385           * Building -> Location -> Latitude (HIS
geb_geogr_breite)
```

```
Hermann-Herder-Straße       * Building -> Address -> Street (HIS street)
10              * Building -> Address -> House number
79104               * Building -> Address -> Postal code (HIS postcode)
Freiburg im Breisgau        * Building -> Address -> City (HIS city)
-               * Building -> Address -> Additional address information (HIS
addressaddition)
45              * Building -> BuildingID -> BuildingID (HIS BuildingID)
```

Beispiel eines Raumes in i-Doit mit der

Zuordnung

```
6200-1021               * Room -> General -> Title (HIS uniquename)
-107                    * Room -> Room -> Room number (HIS shorttext)
1. Kellergeschoß            * Room -> Room -> Floor (HIS geschoss)
Maschinensaal               * Room -> Room -> description (HIS description)
Universitätsrechenzentrum, 1. Kellergeschoß, R -107 (UG)
                    * Room -> General -> Description (HIS longtext)
ADV-Großrechneranl.-raum        * Room -> General -> Category (HIS
raumnutzungsart)
4000202001              * Room -> Accounting -> Account    (HIS
kostenstelle)
RZ Allg. Allgemeiner Geschäftsbetrieb   * room -> Accounting -> Cost unit
(HIS einrichtung)
```

Personen-Objekte in i-Doit

Für bestimmte Aufgaben sollten Personen-Informationen im I-doIT hinterlegt oder verlinkt werden. Dazu wären Überlegungen anzustellen, wie man Ansprechpartner für Netzbereiche (für das Incident-Handling) geeignet im I-doIT hinterlegt. Es ist recht schnell klar, dass es keine gute Idee ist, irgendwelche Personendatensätze dort zu pflegen (da die Daten vermutlich recht schnell auseinanderlaufen würden). Deshalb gingen Überlegungen dahin, das so ähnlich, wie für die Räume umzusetzen und im Personenobjekt einen eindeutigen Identifier zu nutzen, der ins HIS-in-One (Personeninformation) zeigt. Dann findet man dort die aktuelle Telefonnummer und Emailadresse.

Dieses wäre auch für weitere Bereiche, wie Zuordnung von Servern, virtuellen Maschinen, Käufern von Rechnern etc. nützlich. Es gibt ja schon Erfahrungen mit der Nutzung von Webservices des Systems und die Quelle der Anfrage wäre klar abgegrenzt. Man könnte automatisierte Prozesse gegen das HIS laufen lassen, um zu checken, wie lange eine Person noch aktiv ist und so vor Ablauf von deren Vertrag eine Info generieren, dass hier ein potenzieller Wechsel ansteht.

Ziel ist, dass wir möglichst wenig Personendaten im I-doIT haben und diese möglichst leicht aktuell halten können. Deshalb bleibt derzeit die Frage ob HIS-in-One dazu eine gute Instanz ist oder ob man das eher als Zwischenlösung auf dem Weg zum neuen IDM sehen würde.

Organisation der Datenflüsse

Die primäre Datenquelle für die Bestückung der Rauminformationen soll HIS-Bau sein. Wenn zusätzliche Datenfelder, die beim Import nach HIS-in-One erzeugt werden, benötigt werden, sollen diese in den HIS-Bau-Export hinzugefügt werden. Die Details des Workflows werden noch gemeinsam mit Campus-Management besprochen. Hinzu kommen vermutlich noch Personen-Identifier (s.o.) Das Ergebnis wird dann nach Mannheim geschickt, um die Daten in I-doIT zu aktualisieren. Auch hier sind die Details noch zu vereinbaren.

Sicherheit - Artikelübersicht