IT-Sicherheit

Planung | 2021 | Kryptographie und Basismechanismen

Kontaktdaten | Prof. Dr. Carsten Müller

E-Mail carsten.mueller@mosbach.dhbw.de | Tel. (07931) 1230-536 | Skype live:carsten.mueller_1

Modulnummer

T3INF3002 | 48h Präsenzzeit | 102h Selbststudium

► Die geplante Klausur in Präsenzform zu „IT-Sicherheit“ entfällt.

MGH und MOS | 22.02.2021 08:00 - 12:15 Uhr | Online (5 UE)

GotoMeeting | https://global.gotomeeting.com/join/312410981
Organisation | Grundlagen, Klassische Chiffren I

22.02.2021 12:45 - 17:00 Uhr | Begleitetes Selbststudium (5 UE)

MGH und MOS | 25.02.2021 08:00 - 12:15 Uhr | Online (5 UE)

GoToMeeting | https://global.gotomeeting.com/join/668429021
Klassische Chiffren II

25.02.2021 12:45 - 17:00 Uhr | Begleitetes Selbststudium (5 UE)

MGH und MOS | 01.03.2021 08:00 - 12:15 Uhr | Online (5 UE)

GoToMeeting | https://global.gotomeeting.com/join/502600461
Moderne Blockchiffren I und II

01.03.2021 12:45 - 17:00 Uhr | Begleitetes Selbststudium (5 UE)

MGH und MOS | 04.03.2021 08:00 - 12:15 Uhr | Online (5 UE)

GoToMeeting | https://global.gotomeeting.com/join/137931229
Public-Key Kryptographie

04.03.2021 12:45 - 17:00 Uhr | Begleitetes Selbststudium (5 UE)

MGH und MOS | 15.03.2021 09:00 - 12:15 Uhr | Online (4 UE)

GoToMeeting | https://global.gotomeeting.com/join/543111165
Workshop | Q&A (optional)

Alternative Modulprüfung [100 Punkte]

Teil I | Klassische Chiffren [40 Punkte]

Themenbereiche	Aufgaben	Bewertung	Abgabe
Verschiebechiffre	SC01 | hello, k = 4 SC02 | smile, k = 7 SC03 | joker, k = 3 SC04 | virus, k = 9 SC05 | covid, k = 5 SC06 | serum, k = 5	5 Punkte	24.02.2021
Multiplikative Chiffre	Wählen Sie einen geeigneten Wert für t. Zeigen Sie an einem Beispiel die Nutzung eines nicht zulässigen Wertes für t. MC01 | hello; MC02 | smile MC03 | joker; MC04 | virus MC05 | covid; MC06 | serum	5 Punkte	24.02.2021
Affine Chiffre	Wählen Sie einen geeigneten Wert für den Schlüssel k. AC01 | hello; AC02 | smile AC03 | joker; AC04 | virus AC05 | covid; AC06 | serum	5 Punkte	24.02.2021
Vigenere Chiffre	Schlüssel: fycwx VC01 | hello; VC02 | smile VC03 | joker; VC04 | virus VC05 | covid; VC06 | serum	5 Punkte	24.02.2021
Playfair Chiffre	PC01 | Message: fascinating world of cryptography; Key: science PC02 | Message: vaccine is stored in building abc; Key: covid PC03 | Message: bnt162b2 mrna based vaccine ready; Key: biontech PC04 | Message: vaccine is based on messenger rna; Key: curevac PC05 | Message: five single stranded rna molecule; Key: science PC06 | Message: transport starts tomorrow at nine; Key: secret	10 Punkte	27.02.2021
One-Time-Pad	Schlüssel: fycwx OT01 | hello; OT02 | smile OT03 | joker; OT04 | virus OT05 | covid; OT06 | serum	5 Punkte	27.02.2021
Hill Chiffre	Schlüssel: fycwxq; Füllzeichen: z HC01 | hello; HC02 | smile HC03 | joker; HC04 | virus HC05 | covid; HC06 | serum	5 Punkte	27.02.2021

Teil II | Moderne Blockchiffren [10 Punkte]

Themenbereich	Aufgabe	Bewertung	Abgabe
S-DES	Dokumentieren Sie nachvollziehbar die Ver-/Entschlüsselung des Buchstabens "h" mit einem nicht-trivialen 10-Bit-Schlüssel.	10 Punkte	14.03.2021

Teil III | Public-Key Kryptographie [10 Punkte]

Themenbereich	Aufgaben	Bewertung	Abgabe
RSA	Bestimmen Sie den Public Key. Bestimmen Sie den Private Key. A verschlüsselt Nachricht m nach c und kommuniziert c an B. B entschlüsselt die Nachricht c nach m. X attackiert die verschlüsselte Nachricht c, um den Klartext m zu erhalten. PK01|m = hello|p = 3, q = 11, e = 7 PK02|m = smile|p = 5, q = 11, e = 7 PK03|m = joker|p = 7, q = 11, e = 7 PK04|m = virus|p = 3, q = 11, e = 7 PK05|m = covid|p = 5, q = 11, e = 7 PK06|m = serum|p = 3, q = 11, e = 7	10 Punkte	14.03.2021

Aufgabenstellungen | Wichtige Hinweise

- **Bearbeitung** der Teile I-III erfolgt **individuell**.
- **Vorgehensweise** der **Ver-/Entschlüsselung** ist **nachvollziehbar** zu **dokumentieren**.
- Datei **[matrikelnummer]_[task_id].md** im **Format Markdown**[1] ist bis spätestens zum definierten Abgabetermin **in Moodle hochzuladen**.

1 https://dillinger.io/

Mergentheim/Mosbach Security Agency (MSA)

Zu Simulationszwecken wird ein Unternehmensnetzwerk als EventBus dargestellt. An dieses Unternehmensnetzwerk sind fünf Zweigstellen [i] BranchHKG, [ii] BranchCPT, [iii] BranchSFO, [iv] BranchSYD und [v] und BranchWUH angeschlossen.

Zwischen den Zweigstellen werden verschlüsselte Nachrichten ausgetauscht.

Für die Verschlüsselung stehen ein klassisches Verfahren und RSA zur Verfügung.
Die kryptographischen Algorithmen sind in austauschbaren Komponenten gekapselt.
Die Komponenten als jar sind digital signiert.
Nur Komponenten mit valider digitaler Signatur können geladen werden.

Der/die Schlüssel für die Ver-/Entschlüsselung sind in geeigneten Konfigurationsdateien im Format JSON zu speichern.

Bei dem Start der Applikation werden bereits definierte Channel automatisch aufgebaut.

Durch Drücken der **Taste [F3]** wird der **Debug-Modus** aktiviert oder deaktiviert. Bei aktiviertem Debug-Modus wird bei Ausführung der Befehle *encrypt message* und *decrypt message* im Verzeichnis log eine Logdatei [encrypt | decrypt]_[algorithm]_[unix_time_unixseconds].txt angelegt. Diese Logdatei zeichnet die Schritte detailliert und nachvollziehbar auf.

Durch Drücken der **Taste [F8]** wird das **Logfile** mit dem neuesten Zeitstempel in den Ausgabebereich der GUI geladen und angezeigt.

Die Applikation wird über **CQL** (Cryptographic Query Language) gesteuert.

Durch Mausklick auf den **Button execute** in der GUI oder Drücken der **Taste [F5]** wird der Befehl ausgeführt.

- **encrypt message "[message]" using [algorithm] and keyfile [filename]**
  Die zu dem Algorithmus korrespondierende Komponente [algorithm].jar wird dynamisch geladen und die Meldung mit dem key verschlüsselt.
  Die verschlüsselte Meldung wird im Ausgabebereich der GUI angezeigt.
- **decrypt message "[message]" using [algorithm] and keyfile [filename]**
  Die zu dem Algorithmus korrespondierende Komponente [algorithm].jar wird dynamisch geladen und die Meldung mit dem key entschlüsselt.
  Die entschlüsselte Meldung wird im Ausgabebereich der GUI angezeigt.
- **crack encrypted message "[message]" using shift**
  crack encrypted message “[message]” using rsa and keyfile [filename]
  Die zu dem Algorithmus korrespondierende Komponente [shift | rsa]_cracker.jar wird dynamisch geladen und versucht innerhalb von maximal 30 Sekunden die Meldung zu entschlüsseln. Wurde die Meldung innerhalb der Zeitvorgabe entschlüsselt, wird die entschlüsselte Meldung im Ausgabebereich der GUI angezeigt.
  Wurde die Meldung nicht innerhalb der Zeitvorgabe entschlüsselt, erfolgt die Meldung "cracking encrypted message "[message]" failed" im Ausgabebereich der GUI.

- **register participant [name] with type [normal | intruder]**
  Existiert kein Teilnehmer mit diesem Namen wird [i] ein Datensatz in der Tabelle participants und [ii] die Tabelle postbox_[participant_name] angelegt.
  Im Ausgabebereich der GUI wird die Meldung "participant [name] with type [normal | intruder] registered and postbox_[participant_name] created" angezeigt.

  Existiert ein Teilnehmer mit diesem Namen wird die Meldung "participant [name] already exists, using existing postbox_[participant_name]" im Ausgabebereich der GUI angezeigt.

  Für die **Simulation** werden folgende participants angelegt:

branch_hkg	normal
branch_cpt	normal
branch_sfo	normal
branch_syd	normal
branch_wuh	normal
msa	intruder

- **create channel [name] from [participant01] to [participant02]**
  Ein **Channel ist** als **dedizierter EventBus** – basierend auf **Google Guava** – realisiert.

  Für die bidirektionale Kommunikation bzw. das Senden von verschlüsselten Nachrichten zwischen Participant vom Typ normal ist ein Channel notwendig.

  Existiert bereits ein Channel mit dem Namen wird die Fehlermeldung "channel [name] already exists″ im Ausgabebereich der GUI angezeigt. Existiert bereits eine Kommunikationsbeziehung zwischen participant01 und participant02, wird die Fehlermeldung "communication channel between [participant01] and [participant02] already exists" im Ausgabebereich der GUI angezeigt.

  Sind participant01 und participant02 identisch, wird die Fehlermeldung "[participant01] and [participant02] are identical – cannot create channel on itself" im Ausgabebereich der GUI angezeigt.

  Existiert [i] kein Channel mit dem Namen und [ii] keine Kommunikationsbeziehung zwischen den beiden participant, wird ein Datensatz in der Tabelle channel angelegt und die Meldung ″channel [name] from [participant01] to [participant02] successfully created" im Ausgabebereich der GUI angezeigt.

  Nach erfolgreicher Erstellung eines channel erfolgt die Meldung "channel [name] from [participant01] to [participant02] created" im Ausgabebereich der GUI, in der Tabelle channel wird ein Datensatz angelegt.

  Für die **Simulation** werden folgende channel angelegt:

hkg_wuh	branch_hkg	branch_wuh
hkg_cpt	branch_hkg	branch_cpt
cpt_syd	branch_cpt	branch_syd
syd_sfo	branch_syd	branch_sfo

- **show channel**
  Im Ausgabebereich der GUI werden die Channel angezeigt.
  hkg_wuh | branch_hkg and branch_wuh
  hkg_cpt | branch_hkg and branch_cpt
  cpt_syd | branch_cpt and branch_syd
  syd_sfo | branch_syd and branch_sfo

- **drop channel [name]**
  Existiert der channel mit dem Namen, wird dieser Datensatz aus der Tabelle channel gelöscht und im Ausgabebereich der GUI die Meldung "channel [name] deleted" angezeigt. Existiert kein channel mit dem Namen, wird im Ausgabebereich der GUI die Fehlermeldung "unknown channel [name]" ausgegeben.

- **intrude channel [name] by [participant]**
  Existiert der channel mit dem Namen, wird der participant vom Typ intruder für den Channel registriert und erhält alle Nachrichten die über diesen Channel kommuniziert werden.

  Bei Erhalt einer Nachricht wird in der Tabelle postbox des intruder ein neuer Datensatz erstellt, das Attribut message wird auf den Wert unknown gesetzt.

  Der participant lädt dynamisch die zu dem Algorithmus korrespondierende Komponente [algorithm]_cracker.jar und versucht innerhalb von maximal 30 Sekunden die Meldung zu entschlüsseln. Wird innerhalb der Zeitvorgabe die Nachricht erfolgreich entschlüsselt, wird das Attribut message auf den Wert der Meldung im Klartext gesetzt.

  Im Fall einer erfolgreichen Entschlüsselung wird im Ausgabebereich der GUI die Meldung "intruder [name] cracked message from participant [name] | [message]" angezeigt.

  Im Fall einer nicht erfolgreichen Entschlüsselung wird im Ausgabebereich der GUI die Meldung "intruder [name] | crack message from participant [name] failed" angezeigt.

- **send message "[message]" from [participant01] to [participant02] using [algorithm] and keyfile [name]**

  Die Nachricht wird mit dem Algorithmus und keyfile verschlüsselt und von participant01 an participant02 über den Channel kommuniziert.

  Existiert zwischen participant01 und participant02 kein channel, wird die Fehlermeldung "no valid channel from [participant01] to [participant02]" im Ausgabebereich der GUI angezeigt.

  Existiert zwischen participant01 und participant02 ein channel wird die Nachricht mit dem Algorithmus und keyfile verschlüsselt und über den Channel kommuniziert.

  Das Versenden der Nachricht wird in der Tabelle messages protokolliert.

  Der Empfänger participant02 lädt dynamisch die zu dem Algorithmus korrespondierende Komponente [algorithm].jar und entschlüsselt die Nachricht. Nach Entschlüsselung wird in der Tabelle postbox_[participant02_name] ein Datensatz erstellt und Ausgabebereich der GUI die Meldung "[participant02_name] received new message" angezeigt.

algorithms		
id	TINYINT NOT NULL	**PK** Fortlaufende Nummerierung beginnend bei 1
name	VARCHAR(10) NOT NULL	unique

types		
id	TINYINT NOT NULL	**PK** Fortlaufende Nummerierung beginnend bei 1
name	VARCHAR(10) NOT NULL	unique

participants		
id	TINYINT NOT NULL	**PK** Fortlaufende Nummerierung beginnend bei 1
name	VARCHAR(50) NOT NULL	unique
type_id	TINYINT NOT NULL	FK

channel		
name	VARCHAR(25) NOT NULL	**PK**
participant_01	TINYINT NOT NULL	FK ► participants
participant_02	TINYINT NOT NULL	FK ► participants

messages		
id	TINYINT NOT NULL	**PK** Fortlaufende Nummerierung beginnend bei 1
participant_from_id	TINYINT NOT NULL	FK ► participants
participant_to_id	TINYINT NOT NULL	FK ► participants
plain_message	VARCHAR(50) NOT NULL	Meldung im Klartext
algorithm_id	TINYINT NOT NULL	FK ► algorithms
encrypted_message	VARCHAR(50) NOT NULL	Verschlüsselte Meldung
keyfile	VARCHAR(20) NOT NULL	Dateiname keyfile
timestamp	INTEGER	Unix-Zeitstempel in Sekunden

postbox_[participant_name]		
id	TINYINT NOT NULL	**PK** Fortlaufende Nummerierung beginnend bei 1
participant_from_id	TINYINT NOT NULL	FK ► algorithms
message	VARCHAR(50) NOT NULL	Entschlüsselte Meldung
timestamp	INTEGER	Unix-Zeitstempel in Sekunden

Komponente	Schnittstelle
shift.jar	String encrypt(String plainMessage, File keyfile) String decrypt(String encryptedMessage, File keyfile)
shift_cracker.jar	String decrypt(String encryptedMessage)
rsa.jar	String encrypt(String plainMessage, File publicKeyfile) String decrypt(String encryptedMessage, File privateKeyfile)
rsa_cracker.jar	String decrypt(String encryptedMessage, File publicKeyFile)

Anforderung	Bewertung
01 [S1 und S2] Basisarchitektur | 10 Punkte	**09** [S2] send message | 3 Punkte
02 [S1] Taste [F3], Debug-Modus | 2 Punkte	**10** [S1] encrypt message | 3 Punkte
03 [S2] Taste [F8], Logfile | 2 Punkte	**11** [S1] decrypt message | 3 Punkte
04 [S1] register participant | 2 Punkte	**12** [S2] crack message | 6 Punkte
05 [S1] create channel | 2 Punkte	**13** [S1] Komponente shift.jar | 5 Punkte
06 [S2] show channel | 2 Punkte	**14** [S2] Komponente rsa.jar | 5 Punkte
07 [S2] drop channel | 2 Punkte	**15** [S2] Komponente shift_cracker.jar | 10 Punkte
08 [S1] intrude channel | 3 Punkte	**16** [S1] Komponente rsa_cracker.jar | 10 Punkte

Aufgabenstellung | Wichtige Hinweise

- **Bearbeitung** der Komplexaufgabe erfolgt **im Team** durch zwei Studierende.
- **Programmiersprache** | Java 15.0.1 oder höher
- **IntelliJ IDEA** Community 2020.3 oder höher
- Nutzung **GitHub** und Build-Management mit **gradle** 6.7.1 oder höher.
- Datenbank: **HSQLDB** 2.5.1 oder höher.
- **Implementierung** einer technisch einwandfrei lauffähigen Applikation.
  Kommunikation der Nachricht "vaccine for covid is stored in building abc".
- Es sind **keine dedizierten Tests in JUnit** zu erstellen.
- Verwendung geeigneter **englische**r Begriffe für **Namen** und **Bezeichnungen**.
- Erstellung einer **7-Zip-Datei kpl_[matrikelnummern_teammitglieder].7z**
  mit dem vollständigen Projekt und **Upload in Moodle**.
  In der beigefügten **readme.txt** ist die Zuordnung Anforderung/Studierender dokumentiert.
- **Zeitansatz**: 40 Stunden je Studierender S1 und S2
- **Abgabetermin**: Sonntag, 28.03.2021
- **Bewertung**: 40 Punkte je Studierender S1 und S2

Verteilung[2] | Aufgaben zu Themenbereiche I und III

#	Kurs	SC	MC	AC	VC	PC	OT	HC	PK
1044480	MOS-INF19A	SC03	MC01	AC05	VC05	PC02	OT03	HC02	PK02
1103207	MOS-INF19A	SC04	MC04	AC06	VC04	PC02	OT06	HC03	PK05
1253402	MOS-INF19B	SC05	MC05	AC01	VC05	PC05	OT03	HC05	PK04
1633332	MOS-INF19A	SC03	MC06	AC06	VC02	PC05	OT06	HC02	PK03
1705159	MOS-INF19A	SC06	MC05	AC02	VC02	PC04	OT01	HC05	PK05
1716504	MOS-INF19B	SC06	MC03	AC04	VC03	PC02	OT05	HC05	PK01
1724537	MOS-INF19A	SC01	MC02	AC04	VC01	PC01	OT03	HC01	PK05
1959397	MGH-INF19	SC02	MC04	AC01	VC04	PC04	OT05	HC06	PK05
2120099	MOS-INF19A	SC01	MC06	AC02	VC02	PC01	OT05	HC06	PK05
2143592	MGH-INF19	SC05	MC01	AC06	VC05	PC01	OT03	HC04	PK02
2516708	MOS-INF19B	SC03	MC04	AC05	VC06	PC03	OT05	HC03	PK05
2529977	MOS-INF19A	SC05	MC06	AC04	VC04	PC05	OT02	HC01	PK03
2627585	MOS-INF19B	SC05	MC04	AC03	VC01	PC01	OT04	HC06	PK01
2814814	MOS-INF19A	SC04	MC01	AC04	VC06	PC01	OT02	HC03	PK04
2832690	MOS-INF19A	SC01	MC03	AC02	VC04	PC01	OT04	HC03	PK02
2858031	MOS-INF19B	SC04	MC05	AC04	VC01	PC06	OT01	HC03	PK05
2861756	MOS-INF19A	SC02	MC01	AC05	VC05	PC01	OT04	HC01	PK06
3010486	MGH-INF19	SC06	MC05	AC03	VC03	PC03	OT01	HC01	PK02
3106335	MOS-INF19B	SC05	MC01	AC03	VC04	PC01	OT01	HC05	PK02
3110300	MOS-INF19A	SC04	MC06	AC04	VC05	PC03	OT05	HC02	PK04
3186523	MOS-INF19B	SC02	MC05	AC01	VC06	PC06	OT02	HC03	PK06
3326612	MOS-INF19B	SC03	MC02	AC06	VC02	PC04	OT05	HC06	PK06
3389310	MOS-INF19B	SC01	MC06	AC06	VC03	PC06	OT05	HC06	PK06
3407192	MOS-INF19A	SC03	MC04	AC04	VC04	PC03	OT02	HC05	PK04
3932085	MGH-INF19	SC04	MC01	AC02	VC06	PC02	OT02	HC05	PK03
3939573	MGH-INF19	SC02	MC03	AC01	VC05	PC04	OT06	HC02	PK05
3980329	MOS-INF19B	SC05	MC03	AC03	VC03	PC04	OT02	HC03	PK04
3994729	MGH-INF19	SC06	MC04	AC02	VC04	PC02	OT05	HC02	PK01
4002027	MOS-INF19A	SC03	MC03	AC04	VC05	PC01	OT03	HC02	PK05
4085242	MOS-INF19A	SC01	MC05	AC05	VC04	PC02	OT05	HC03	PK03
4153197	MGH-INF19	SC06	MC02	AC05	VC02	PC03	OT02	HC04	PK04
4485500	MOS-INF19B	SC04	MC04	AC06	VC04	PC03	OT01	HC01	PK02
4591230	MGH-INF19	SC01	MC06	AC03	VC03	PC03	OT06	HC04	PK03
4669114	MOS-INF19B	SC02	MC02	AC02	VC06	PC04	OT02	HC04	PK02
4670401	MOS-INF19B	SC01	MC01	AC03	VC03	PC05	OT02	HC04	PK01
4775194	MOS-INF19A	SC04	MC05	AC04	VC01	PC05	OT04	HC05	PK04
4834957	MOS-INF19A	SC04	MC05	AC04	VC01	PC03	OT06	HC01	PK06
5202059	MOS-INF19B	SC02	MC03	AC02	VC06	PC05	OT06	HC02	PK05
5404118	MOS-INF19A	SC06	MC05	AC06	VC02	PC05	OT04	HC02	PK06
5542852	MGH-INF19	SC04	MC01	AC05	VC02	PC03	OT06	HC05	PK02
5663902	MGH-INF19	SC03	MC04	AC02	VC02	PC03	OT03	HC01	PK03
5703004	MOS-INF19B	SC01	MC01	AC05	VC05	PC06	OT06	HC03	PK04
5736465	MOS-INF19B	SC06	MC02	AC01	VC06	PC06	OT04	HC05	PK04
5807262	MOS-INF19A	SC05	MC02	AC05	VC04	PC01	OT06	HC03	PK01

2 https://www.random.org/lists/ | 22.12.2020

#	Kurs	SC	MC	AC	VC	PC	OT	HC	PK
5813630	MOS-INF19A	SC01	MC05	AC05	VC04	PC06	OT03	HC04	PK01
5986488	MOS-INF19B	SC05	MC06	AC06	VC05	PC04	OT03	HC04	PK05
6039197	MGH-INF19	SC04	MC04	AC03	VC02	PC05	OT02	HC06	PK02
6048166	MOS-INF19B	SC02	MC03	AC01	VC03	PC03	OT04	HC02	PK02
6089394	MGH-INF19	SC06	MC04	AC05	VC05	PC04	OT05	HC04	PK04
6143217	MOS-INF19A	SC01	MC02	AC05	VC01	PC03	OT01	HC06	PK04
6196929	MOS-INF19B	SC06	MC06	AC03	VC03	PC06	OT01	HC04	PK06
6217046	MGH-INF19	SC06	MC04	AC06	VC02	PC01	OT02	HC06	PK03
6288954	MGH-INF19	SC03	MC06	AC01	VC02	PC05	OT02	HC06	PK01
6499887	MOS-INF19A	SC01	MC03	AC06	VC01	PC04	OT06	HC05	PK06
6558328	MGH-INF19	SC01	MC02	AC03	VC03	PC06	OT01	HC02	PK03
6698461	MGH-INF19	SC03	MC01	AC02	VC02	PC06	OT06	HC03	PK04
6969415	MGH-INF19	SC02	MC05	AC03	VC06	PC05	OT03	HC01	PK02
7008808	MOS-INF19A	SC02	MC06	AC02	VC01	PC06	OT04	HC05	PK05
7089612	MGH-INF19	SC06	MC06	AC03	VC05	PC01	OT01	HC01	PK03
7631677	MGH-INF19	SC01	MC01	AC01	VC01	PC02	OT03	HC02	PK06
7862288	MOS-INF19A	SC05	MC04	AC04	VC01	PC02	OT01	HC01	PK01
7903471	MOS-INF19A	SC05	MC03	AC06	VC06	PC05	OT03	HC04	PK06
8438008	MGH-INF19	SC05	MC03	AC05	VC03	PC02	OT05	HC02	PK02
8622410	MOS-INF19A	SC01	MC03	AC02	VC05	PC04	OT02	HC06	PK01
8843476	MOS-INF19B	SC03	MC01	AC01	VC02	PC03	OT04	HC06	PK03
8864957	MOS-INF19A	SC03	MC02	AC06	VC01	PC02	OT04	HC01	PK03
8905135	MGH-INF19	SC05	MC05	AC02	VC03	PC04	OT04	HC03	PK04
9008480	MOS-INF19B	SC02	MC03	AC05	VC01	PC04	OT06	HC02	PK06
9217288	MGH-INF19	SC06	MC01	AC04	VC06	PC06	OT05	HC06	PK01
9282087	MGH-INF19	SC03	MC02	AC03	VC01	PC01	OT01	HC04	PK01
9295660	MOS-INF19A	SC02	MC03	AC04	VC04	PC02	OT04	HC01	PK02
9514094	MOS-INF19A	SC02	MC06	AC02	VC03	PC05	OT01	HC04	PK01
9668368	MOS-INF19B	SC04	MC04	AC04	VC03	PC06	OT03	HC05	PK06
9783115	MGH-INF19	SC03	MC02	AC01	VC04	PC02	OT03	HC03	PK03
9804523	MGH-INF19	SC02	MC05	AC06	VC06	PC02	OT04	HC01	PK03
9899545	MOS-INF19A	SC04	MC01	AC03	VC06	PC04	OT01	HC04	PK01