Bundesverband der Deutschen Volksbanken und Raiffeisenbanken e. V.
Bundesverband deutscher Banken e. V.
Bundesverband Öffentlicher Banken Deutschlands e. V.
Deutscher Sparkassen- und Giroverband e. V.
Verband deutscher Pfandbriefbanken e. V.

Die Deutsche Kreditwirtschaft

Anlage 2

der Schnittstellenspezifikation für die Datenfernübertragung zwischen Kunde und Kreditinstitut gemäß DFÜ-Abkommen

„Spezifikation Echtzeitbenachrichtigungen“

Version 1.0 vom 17.07.2019

Final Version

Inhaltsverzeichnis

.

1 Einführung

Im Zusammenhang mit der Einführung von Echtzeitüberweisungen (Instant Payments) zeigt sich, dass die Möglichkeit, Zahlungen in Echtzeit abzuwickeln nicht nur ein wesentlicher Komfortgewinn für Privatpersonen beispielsweise durch Bezahlmöglichkeiten mit einer Smartphone-App ist, sondern gerade auch für Firmenkunden immense Vorteile bietet.

Während Zahlprozesse noch überwiegend als zeitraubende Batch-Prozesse mit zu beachtenden Cut-Off Zeiten ausgeführt werden, sind doch heute schon die meisten ERP-Systeme Real-Time-Systeme. Neben der Finalität einer Zahlung in Echtzeit ist es die unmittelbare Rückmeldung über den Zahlungserfolg, der für Firmenkunden besonders attraktiv ist und auf dessen Basis weitere Aktionen in einer Geschäftsprozesskette ausgelöst werden können (wie z. B. Lieferung einer Ware). Hierfür ist es aber unerlässlich, dass nicht nur die Zahlung in Echtzeit abgewickelt wird, sondern dass die Benachrichtigung über den Zahlungserfolg auch in Echtzeit erfolgt. Hierfür muss für Firmenkunden eine Lösung geschaffen werden.

In der Regel nutzen Firmenkunden im elektronischen Zahlungsverkehr mit Banken den EBICS-Standard, der von allen Banken und Sparkassen unterstützt wird und eine sichere Übertragung von Zahlungen und Kontoinformationen über das Internet gewährleistet. Als Client-Server-Verfahren konzipiert geht die Initiative zur Übertragung immer von der Kunden- bzw. der Client-Seite aus. Kunden schicken Zahlungen und/oder holen Daten ab. Die Bankseite schickt niemals initiativ Daten an den Kunden. Für die gewünschte Echtzeit-Benachrichtigung von Kunden im Zusammenhang mit Instant Payments bedeutet dies, dass Kunden über EBICS aktiv nachfragen müssten, ob entsprechende Informationen zur Abholung über EBICS bereitstehen. Dies führt unweigerlich zu der Situation, dass EBICS-Kundensysteme in regelmäßigen kurzen Zeitabständen die Bank-Server abfragen (Polling), was zur Überlastung der Banksysteme aufgrund des hohen und dabei überflüssigen Kommunikations-Overheads führen kann. Um dies zu verhindern, könnte es eine Lösung sein, EBICS derart zu verändern, dass über EBICS aktiv Informationen (Push-Service) an Kunden übertragen werden können. Eine solche Lösung würde aber bedeuten, dass die Banken aktiv EBICS-Verbindungen in die Kundensphäre aufbauen müssten, was eine Fülle von zu lösenden Sicherheitsfragen nach sich ziehen würde und zu einer komplexen bankseitigen Administration von EBICS-Kundensystemen führt.

Um einerseits eine komplexe und grundlegende Änderung des EBICS-Standards zu vermeiden und andererseits zu verhindern, dass vermehrt institutsindividuelle Lösungen zur Übermittlung von Echtzeitinformationen geschaffen werden, die nicht miteinander kompatibel sind, hat sich die Deutsche Kreditwirtschaft dazu entschlossen, eine Standardisierung auf Basis der Kombination zweier systemisch unabhängiger Verfahren zu schaffen. Hierzu soll es einerseits bei der Nutzung des sicheren und etablierten EBICS-Verfahrens zur Abholung von bankfachlichen Daten bleiben, zum andern aber moderne Webtechnologie genutzt werden, um die EBICS-Abholung durch den Kunden eventgesteuert triggern zu können. Hierzu wird der etablierte Internetstandard „WebSocket“ genutzt, der einen von der Clientseite aufgebauten und mittels TLS gesicherten permanenten birektionalen Kanal zwischen Firmenkunde und Bank bietet, über den die Banken die Verfügungstellung (Bereitstellung zur Abholung) von Echtzeitinformationen avisieren können. Kundenseitig können dann die bankfachlich relevanten Echtzeitinformationen über das bewährte EBICS-Protokoll sicher zum Kunden übertragen werden.

Der oben beschriebene Lösungsansatz bietet ein Zusammenspiel von etablierten Internetstandards mit dem in Europa etablierten Electronic-Banking-Standard EBICS zur sicheren Übermittlung von Echtzeitbenachrichtigungen an Firmenkunden. In erster Linie ist dieses Verfahren für die Übermittlung des Haben-Avis vorgesehen, welches den Firmenkunden in Echtzeit über den Erfolg einer Echtzeitzahlung informiert. Das Verfahren ist aber darüber hinaus geeignet, auch weitere zukünftige Geschäftsprozesse zu unterstützen, die die Übermittlung von Echtzeitbenachrichtigungen erfordern.

Das Design der im Folgenden näher spezifizierten Lösung bietet aufgrund der systemischen Trennung von EBICS- und WebSocket-Funktionalität grundsätzlich die Möglichkeit der Nutzung vorhandener IT-Infrastrukturen sowohl in der Bank- als auch in der Kundensphäre. Die sichere EBICS-Infrastruktur wird hierbei unverändert für die Übermittlung der sensiblen bankfachlichen Daten genutzt.

2 Technische Architektur des Verfahrens

2.1 Nutzung des WebSocket-Protokolls für Echtzeitbenachrichtigungen

Für die Möglichkeit, serverseitig Nachrichten an einen Client zu schicken (Push-Funktionalität) wird das wss-Protokoll verwendet.
Die konkrete Beschreibung des wss-Protokolls findet sich in RFC 6455[1].

Während der Server bei einer https-Verbindung auf Anfragen eines Clients reagiert (Client Request / Server Response), öffnet der Client beim WebSocket-Protokoll die (bzw. eine) Verbindung zum Server. Der Server kann dann diese offen bleibende Verbindung dazu verwenden, aktiv (d.h. ohne einen konkreten Request des Clients) Informationen an den Client zu liefern. Das Verfahren ist grundsätzlich bidirektional, d.h. der Client kann auch Antworten senden. Für den hier beschriebenen Anwendungsfall werden allerdings nur Nachrichten vom Server an den Client geschickt.
Zur sicheren Datenübertragung wird die Internetverschlüsselung TLS verwendet (wss-Verbindung).

Die Anzahl der Sessions (Verbindungen) eines Kunden mit seiner Bank ist grundsätzlich nicht beschränkt. Mehrere Sessions sind zum Beispiel bei getrennten Kundensystemen erforderlich. Die Mitteilungen sind jedoch kundenbezogen, d.h. bei mehreren Verbindungen zwischen einem Kunden und einer Bank werden Nachrichten auch mehrfach zum Kunden übertragen. Nachrichten an einen Kunden werden immer an die zum Zeitpunkt des Versands bestehenden WebSocket-Verbindungen des Kunden geschickt. Sollte ein Kunde zum Zeitpunkt des Versands einer Nachricht keine geöffnete Verbindung haben, dann entscheidet die Bank, ob sie aufgelaufene Pushnachrichten sendet, sobald unterbrochene Verbindungen wieder verfügbar sind.

[1] https://tools.ietf.org/html/rfc6455

2.2 Daten für den clientseitigen Verbindungsaufbau

Um eine wss-Verbindung zum Server aufzubauen, müssen dem Client die notwendigen Verbindungsdaten mitgeteilt werden.

Eine Möglichkeit für EBICS-Clients ist das Abholen der Daten per EBICS-Download-Standardrequest (ohne Angabe eines von-bis-Datums). Den Banken steht es aber prinzipiell frei, die Zugangsdaten auch über alternative Kanäle den Kunden mitzuteilen. Für den betrachteten initialen Anwendungsfall handelt es sich jedoch um EBICS Kunden, daher ist die Nutzung der sicheren EBICS-Kommunikation zur Übermittlung von Zugangsdaten sinnvoll. Als EBICS-Response erhält der Client die Nutzdaten im JSON-Format.

Es wird folgender EBICS- Geschäftsvorfall definiert:

EBICS V 3.0 / BTF:
Die Belegung der Elementgruppe <Service> ist wie folgt:

Service-Name	Scope	Option	MsgNm	Container	Beschreibung	Up-/Download
OTH	DE		wssparam		Abholen von Daten zum Verbindungsaufbau einer wss-Session im JSON-Format	D

EBICS V 2.5 / fachliche Auftragsart:
Es wird die fachliche Auftragsart WSS verwendet:

Auftragsart	Richtung	Beschreibung	Format
WSS	D	Abholen von Daten zum Verbindungsaufbau einer wss-Session	JSON-Format

Die Response enthält (unabhängig von der verwendeten EBICS-Version) unter
`ebicsResponse/body/DataTransfer/OrderData`
eine JSON-Datei mit folgenden Inhalten, die gemäß EBICS-Spezifikation gezippt und verschlüsselt wird:

Name	Beschreibung	Erläuterungen	Kardinalität
URL	URL des Zugangs		[1..1]

Name	Beschreibung	Erläuterungen	Kardinalität
TOKEN	Sicherheitstoken	Im Zeitraum der Gültigkeit verwendbares Passwort für den wss-Verbindungsaufbau. Anforderung ist, dass dieser pro Banksystem eindeutig ist. Es wird empfohlen, eine UUID (Version 4, d.h. Zufallszahl) gemäß RFC 4122[2] mit der Länge von 16 Bytes zu verwenden.	[1..1]
OTT	One-Time-Token	Kann dieser Token nur einmal (Wert: Y) oder auch mehrfach (Wert: N) verwendet werden?	[1..1]
VALIDITY	Gültigkeitsende zum Aufbau der wss-Verbindung mittels Zugangstoken	Bei Abbruch der Verbindung kann diese mit dem Sicherheitstoken erneut gestartet werden, so lange das Gültigkeitsende nicht erreicht ist. Das Format ist ISO DateTime mit Zeitzone UTC.	[1..1]
PARTNERID	Kundenreferenz	Für EBICS-Kunden empfiehlt sich hier die EBICS-Kunden-ID.	[1..1]
USERID	User-Referenz	Für EBICS-Kunden empfiehlt sich hier die jeweilige EBICS-Teilnehmer-ID. Im Falle eines Requests durch eine SystemID (technischer User), wird hier – soweit vorhanden - die Teilnehmer-ID eines bankfachlichen Users angegeben, andernfalls die SystemID.	[0..1]

Alle Werte in der JSON-Datei sind Zeichenketten.

Beispiel:

```
{
  "URL": "https://bankmitwebsocket.de",
    "TOKEN": "550e8400-e29b-11d4-a716-446655440000",
  "OTT": "N",
  "VALIDITY": "2019-03-21T10:35:22Z",
  "PARTNERID": "K1234567",
  "USERID": "USER4711"
}
```

[2] https://tools.ietf.org/html/rfc4122

Es sind die für einen Standard-Download üblichen EBICS-Fehlercodes möglich. Insbesondere ist dieser Geschäftsvorfall nicht für den „historischen“ Download eingerichtet.

2.3 Headerbeschreibung für den https-Handshake

Prämisse: Auf Grund der hohen Kompatibilität mit den verschiedenen Programmiersprachen (Java, Javascript, Ajax,..) und High Level Frameworks wie Spring-Boot, sollte bei der Übermittlung der Credentials auf http-Basic Authentication RFC 2617[3] zurückgegriffen werden.

Dort wird der folgende Header definiert:
Authorization: Basic <base64encoded Credential>

Das Credential besteht üblicherweise aus User:Passwort und wird daher hier aus den Key-Values von PARTNERID, ggf. USERID und TOKEN in der Form PARTNERID_USERID:TOKEN gebildet (vergl. Kapitel 2.2).

In dem Fall, dass nur eine Kundenreferenz verwendet wird, wird auch das Trennzeichen "_" weggelassen.

Bei dem Beispiel aus Kapitel 2.2 ergibt sich für den Header:

- ➔ Credential = K1234567_USER4711:550e8400-e29b-11d4-a716-446655440000
- ➔ base64encoded Credential= SzEyMzQ1NjdfVVNFUjQ3MTE6NTUwZTg0MDAtZTI5Yi0xMWQ0LWE3MTY-tNDQ2NjU1NDQwMDAw
- ➔ **Authorization: Basic SzEyMzQ1NjdfVVNFUjQ3MTE6NTUwZTg0MDAtZTI5Yi0xMWQ0LWE3MTY-tNDQ2NjU1NDQwMDAw**

[3] https://tools.ietf.org/html/rfc2617

3 Struktur von Echtzeitbenachrichtigungen

Echtzeitbenachrichtigungen, die innerhalb einer wss-Session an den Kunden geliefert werden, sind im JSON-Format kodiert.

Zu unterscheiden sind zwei Klassen von Nachrichten:

1. Die Avisierung neuer Daten auf dem EBICS-Bankserver sowie
2. Allgemeine Informationen (Broadcast-Nachrichten) für Kunden

Die Inhalte und Struktur im JSON-Format sind in den folgenden Unterkapiteln beschrieben. Alle Werte in der JSON-Datei sind Zeichenketten (UTF-8).

3.1 Avisierung neuer Daten auf dem EBICS-Bankserver

Ebene	Name	Beschreibung	Erläuterungen	Kardinalität
1	MCLASS	Nachrichtenklasse als Array mit den Schlüsselinformationen NAME VERS TIMESTAMP		[1..1]
2	NAME	Name der Nachrichtenklasse	Für die hier vorliegende Klasse immer „EBICS-HAA“ *Mit der administrativen EBICS-Auftragsart HAA könnte man die gleichen Infos auch abholen – die JSON-Datei ist demnach das Push-Pendant zum entsprechenden administrativen EBICS-Geschäftsvorfall*	[1..1]
2	VERS	Formatversion der vorliegenden Nachrichtenklasse	Diese Spezifikation startet für Klasse „EBICS-HAA“mit „1.0“	[1..1]
2	TIMESTAMP	Zeitpunkt der Lieferung	Im dem Fall, dass keine aktive wss-Verbindung zum Kunden verfügbar ist, die Daten jedoch bei Verfügbarkeit noch nachgeliefert werden sollen, ist dies der Zeitpunkt des ersten Bereitstellungsversuches der Bankseite Das Format ist ISO DateTime mit Zeitzone UTC	[1..1]
1	PARTNERID	EBICS-Kundenkennung	Die je Bank eindeutige EBICS-Identifikation des Kunden auf dem EBICS-Bankrechner	[1..1]

Ebene	Name	Beschreibung	Erläuterungen	Kardinalität
1	USERID	EBICS-Teilnehmer-kennung	Die pro Firmenkunde eindeutige EBICS-Identifikation des Teilnehmers auf dem EBICS-Bankrechner	[0..1]
1	BTF	Aufführung von BTF-Parameter-gruppen als Arrays	BTF-Parametergruppen, zu denen Informationen zur Abholung (per EBICS-Version 3.0) bereitstehen.	[0..1]
2	SERVICE	Die Namen für diese Schlüsselwörter sind an die entsprechenden Datenelement-/ Attribut-Namen aus der EBICS-Spezifikation angelehnt.	Werte für diese Schlüssel ergeben sich aus der EBICS-Spezifikation (insbesondere externe BTF-Codeliste)	[1..1]
2	SCOPE			[0..1]
2	OPTION			[0..1]
2	CONTTYPE			[0..1]
2	MSGNAME			[1..1]
2	VARIANT			[0..1]
2	VERSION			[0..1]
2	FORMAT			[0..1]
1	ORDERTYPE	Aufführung von Auftragsarten als Array	Auftragsarten, zu denen Informationen zur Abholung (per EBICS-Version 2.5) bereitstehen. [“OrderType1”,”OrderType2”,…”OrderTypeN”]	[0..1]

Hinweis zur Kardinalität von BTF und ORDERTYPE: Mindestens eines dieser optionalen Arrays muss bei der Nachrichtenklasse EBICS-HAA geliefert werden!

Beispiel 1:
Avis eines Echtzeitüberweisungseingangs

```
{
"MCLASS":
[
{"NAME": "EBICS-HAA",
 "VERS": "1.0",
 "TIMESTAMP": "2019-05-13T12:21:50Z"}
],
 "PARTNERID": "K1234567",
 "USERID": "USER471",
 "BTF":
[
{"SERVICE": "REP",
 "SCOPE": "DE",
 "CONTTYPE": "ZIP",
 "MSGNAME": "camt.054"
}
],
 "ORDERTYPE":["C5N"]
}
```

Beispiel 2:
Neue Vormerkposten und ein Payment Status Report zu Echtzeitüberweisungen des Kunden

```
{
"MCLASS":
[
{"NAME": "EBICS-HAA",
 "VERS": "1.0",
 "TIMESTAMP": "2019-05-13T12:21:53Z"}
],
 "PARTNERID": "K1234567",
 "USERID": "USER471",
 "BTF":
[
{"SERVICE": "REP",
 "SCOPE": "DE",
 "CONTTYPE": "ZIP",
 "MSGNAME": "camt.052"
},
{"SERVICE": "REP",
 "SCOPE": "DE",
 "OPTION": "SCI",
 "CONTTYPE": "ZIP",
 "MSGNAME": "pain.002"
}
],
 "ORDERTYPE":["C52","CIZ",]
}
```

3.2 Allgemeine Informationen

Ebene	Name	Beschreibung	Erläuterungen	Kardinalität
1	MCLASS	Nachrichtenklasse als Array mit den Schlüsselinformationen NAME VERS TIMESTAMP		[1..1]
2	NAME	Name der Nachrichtenklasse	Für die hier vorliegende Klasse immer „INFO“	[1..1]
2	VERS	Formatversion der vorliegenden Nachrichtenklasse	Diese Spezifikation startet für Klasse „INFO“ mit „1.0“	[1..1]

Ebene	Name	Beschreibung	Erläuterungen	Kardinalität
2	TIMESTAMP	Zeitpunkt der Lieferung	Im dem Fall, dass keine aktive wss-Verbindung zum Kunden verfügbar ist, die Daten jedoch bei Verfügbarkeit noch nachgeliefert werden sollen, ist dies der Zeitpunkt des ersten Bereitstellungsversuches der Bankseite Das Format ist ISO DateTime mit Zeitzone UTC	[1..1]
1	INFO	Freitextnachrichten als Array mit den Schlüsselinforma-tionen LANG FREE	Das Array muss mindestens ein Schlüsselpaar LANG / FREE enthalten.	[1..1]
2	LANG	Sprache des Freitextes	Angabe des zweistelligen Länderkürzels	[1..1]
2	FREE	Freitextnachricht	Die Mitteilung kann theoretisch beliebig lang sein	[1..1]

Beispiel 3:

Beispiel für eine allgemeine Information

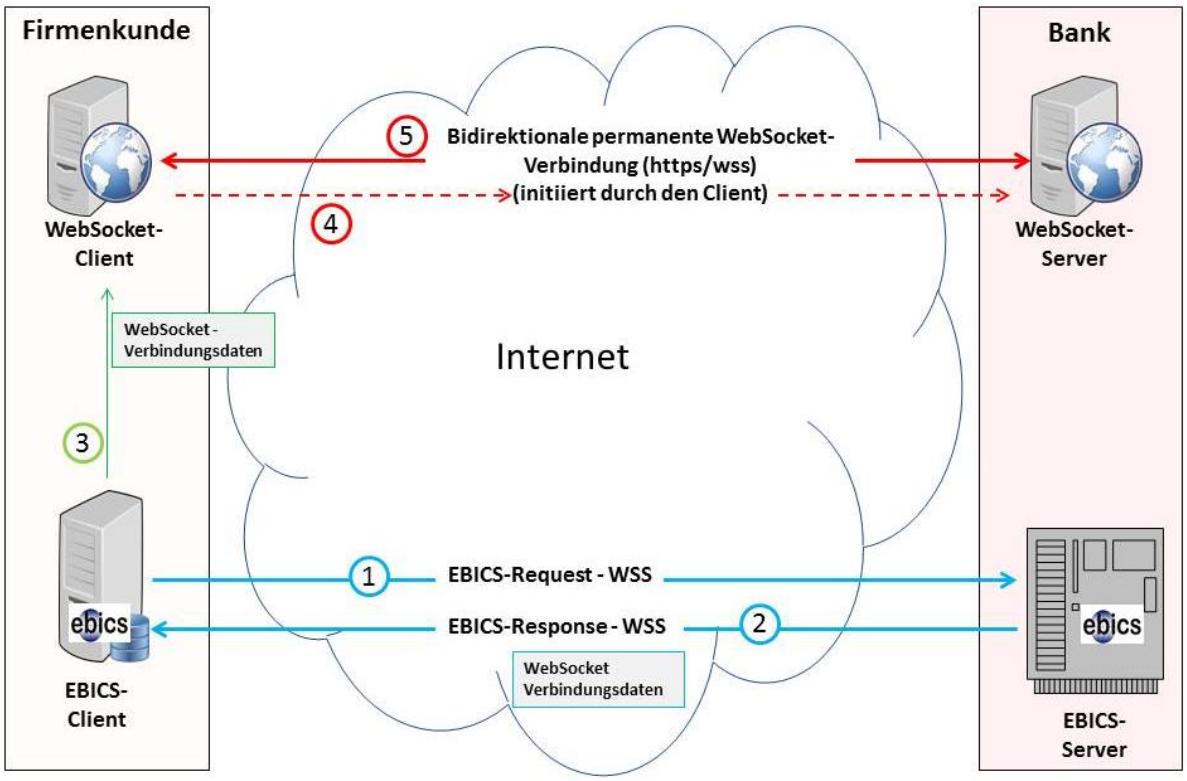
```
{
"MCLASS":
[
{"NAME": "INFO",
 "VERS": "1.0",
 "TIMESTAMP": "2019-03-25T12:25:34Z"}
],
 "INFO":
[
{"LANG": "DE",
 "FREE": "Der EBICS-Service ist am 30.03.2019 von 10:00 – 11:00 Uhr wegen
Wartungsarbeiten nur eingeschränkt verfügbar"
}
]
}
```

4 Abgrenzung EBICS Sphäre – WebSocket-Verbindung

Über die WebSocket-Verbindung kann ein EBICS-Kunde in Echtzeit über neu auf dem EBICS-Bankserver bereistehenden Dateien informiert werden (vergleiche Kapitel 3.1).

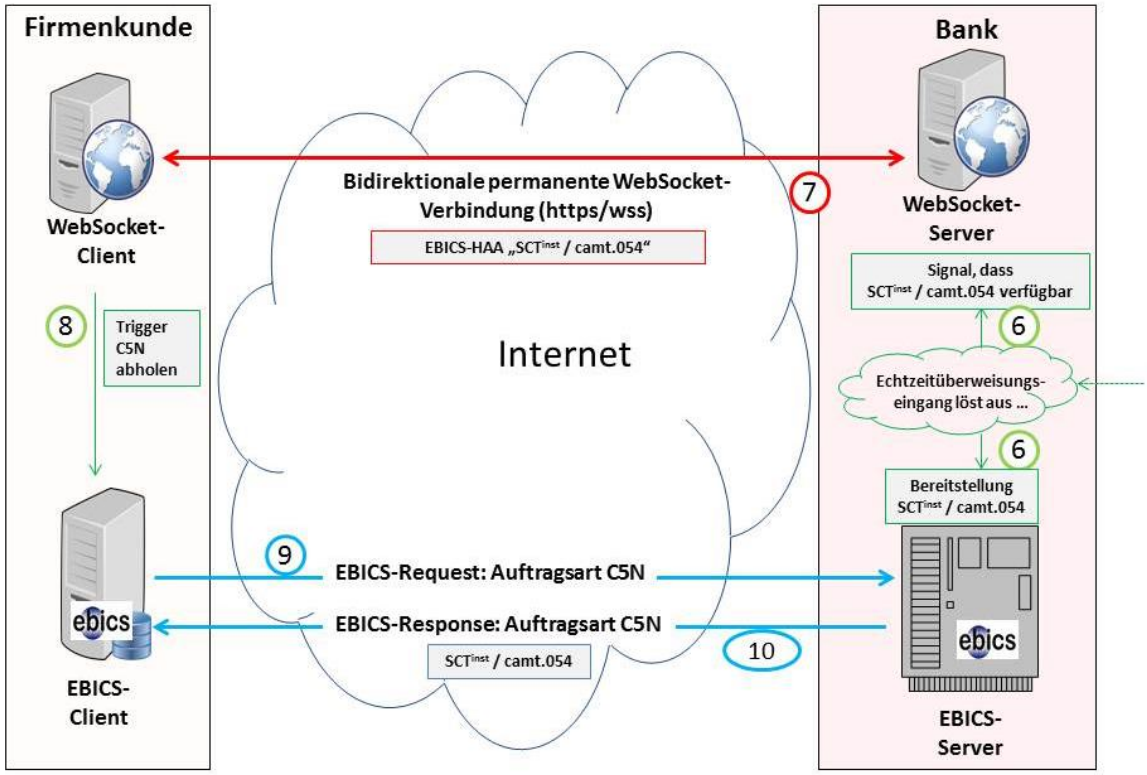
Die WebSocket-Verbindung ist somit eine Ergänzung der EBICS-Kommunikation, ist allerdings vom EBICS-Verfahren technisch vollkommen getrennt.

Der Aufbau der WebSocket-Verbindung ist in der folgenden Grafik mit den einzelnen Aktionen nach zeitlichem Ablauf durchnummeriert und für die verschiedenen Sphären in unterschiedlichen Farben dargestellt (blau = EBICS, rot = WebSocket, grün = Zusammenspiel WebSocket und EBICS in der Kundensphäre):



Die folgende Grafik zeigt als Beispiel einer Echtzeitbenachrichtigung die Avisierung eines Echtzeitüberweisungseingangs.

Auch hier sind die einzelnen Aktionen nach zeitlichem Ablauf durchnummeriert und für die verschiedenen Sphären in unterschiedlichen Farben dargestellt (blau = EBICS, rot = WebSocket, grün = Zusammenspiel WebSocket und EBICS in der Kundensphäre bzw. in der Banksphäre):



4.1 Aktionen auf Bankseite

Die Bank, die den Echtzeitbenachrichtigungsservice anbieten möchte, muss die Auftragsart WSS bzw. die entsprechenden BTF-Parameter anbieten.
Die wss-Zugangs-Daten müssen im beschriebenen JSON-Format erstellt werden können, um sie dann durch den EBICS-Server bereitzustellen.
Die Bankseite muss die N potenziellen WebSocket-Verbindungen für ihre M Kunden verwalten können. Die entsprechenden Anmeldetoken müssen ebenfalls verwaltet werden, insbesondere muss ein Abgleich mit den zugeordneten Kunden-IDs erfolgen.
Zudem erstellt die Bank (neartime) JSON-Nachrichten, wenn neue Daten auf dem EBICS-Server verfügbar sind oder allgemeine Informationen schnell an den Kunden gesendet werden sollen.

4.2 Aktionen auf Kundenseite

Ein Kunde, der Echtzeitbenachrichtigungen erhalten möchte, muss die Auftragsart WSS bzw. die entsprechenden BTF-Parameter durchführen können.
Insbesondere muss sein System die zurückgelieferte JSON-Struktur aus der EBICS-Response verstehen und daraus eine entsprechende WebSocket-Verbindung aufbauen können.
Zudem muss das Kundensystem die Push-Nachrichten aus der WebSocket- Verbindung verstehen und entsprechende Aktionen ableiten können (insbesondere Trigger für entsprechende Download-Requests in EBICS).