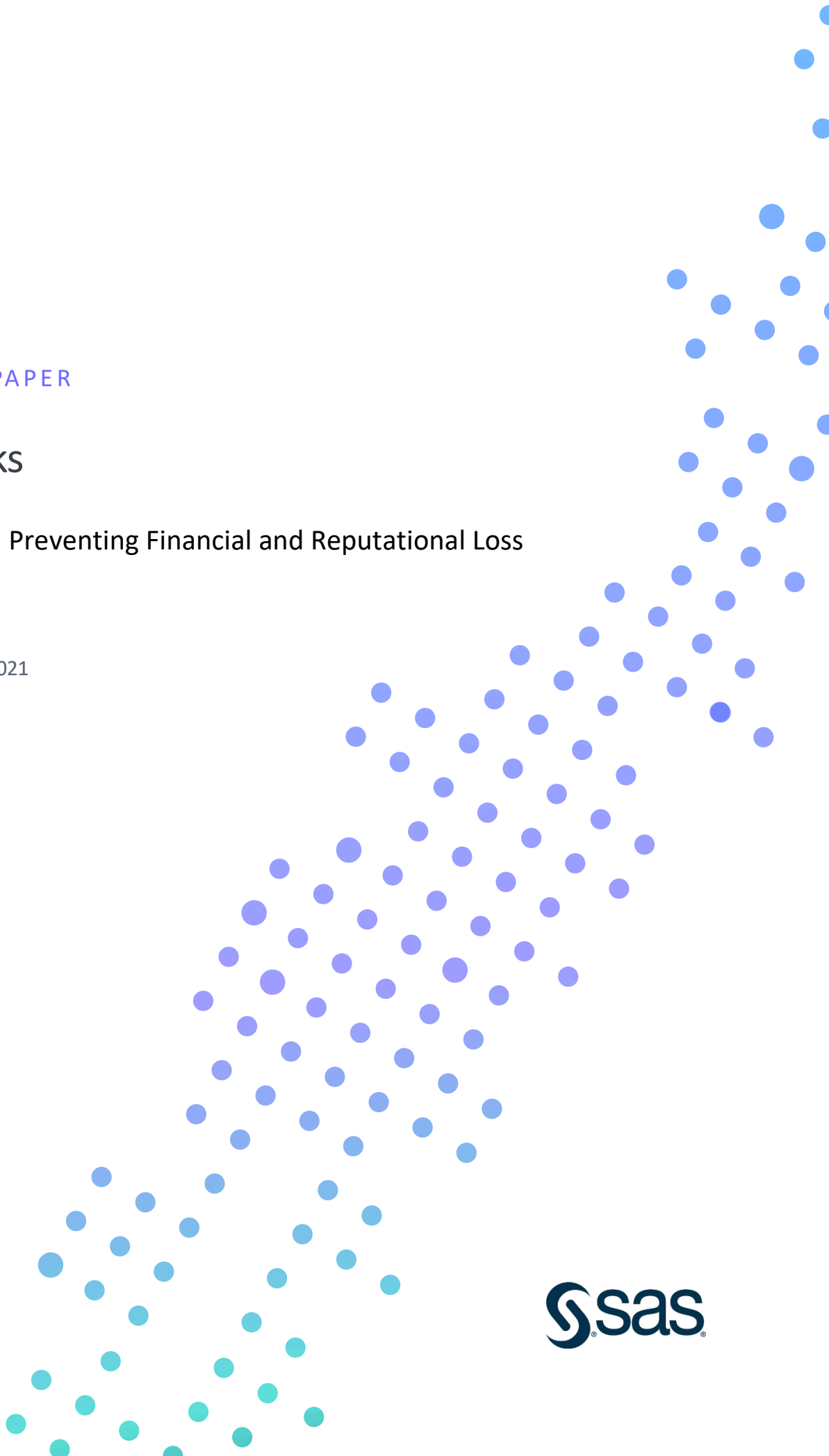
TECHNICAL PAPER

BIN Attacks

Predicting and Preventing Financial and Reputational Loss

Last update: April 2021



sas

Contents

Relevant Products and Releases

- SAS® Fraud Management 6.1

Overview

It is not rocket science to use a simple program to predict a valid credit card's primary account number (PAN). Thanks to the internet, many online bank identification number (BIN) verification databases are freely available where anyone can access and validate a given BIN. Using a program and the BIN database, fraudsters can effortlessly predict a valid PAN for a credit card and to use this number fraudulently for financial gain. With the rise of online shopping, e-commerce offers a low-risk opportunity for people who are perpetrating the fraud. After all, this type of fraud eliminates face-to-face contact and supports higher volumes of transactions at a much lower cost because there are no requirements to produce a fake plastic card to commit fraudulent activities.

Figure 1. Credit Card Numbers

Other than the PAN, the values that are needed to complete an online transaction are the expiration (expiry) date and the security code (the code on the signature panel). Unfortunately, there are merchants who still do not enforce the need for the security code, which leaves only the expiration date.

Expiration dates are specific to the month and year only. Unfortunately, this is where banks hurt themselves. Banks have historically issued cards in sequence, so if fraudsters know the expiration date for one card, they know the expiration date is the same for the rest of the cards issued during that same month! All the fraudster needs to do is find the upper and lower boundaries for the card number range for each month.

It is the knowledge of the boundaries that results in a BIN attack. A bot is used to iterate through the card number range to issue low- value payments to the same merchant. Eventually, card numbers begin to be consistently blocked. One declined transaction could be due to cancellation or reissue of a specific card. However, a consistent range of subsequently declined transactions means that the fraudster can be confident that the expiration date has changed.

What Is The Modus Operandi?

There are only a few steps to trigger a BIN attack:

- Gather a genuine BIN and expiration date from an internet compromise or from a discarded credit card transaction receipt. (The BIN is not typically masked.)
- Generate the remaining 5 to 9 digits (the unique identifier of an account) with a program to create the required PAN. The PAN is typically 12 to 16 digits.
- The last digit of the PAN is a checksum and there is mod-10 logic to predict this number. (This logic is not covered in this paper.)
- With steps 1-3, the fraudster creates a list of PANs. Combine these numbers with the expiration date.
- Use a bot to process authorizations via an online merchant for a small value purchase (around USD 1). This purchase confirms that the PAN and expiration date combination is valid.
- Bingo! Start to purchase high-value goods.

From the preceding steps, you can see that an attempted BIN attack involves minimal effort and risk from a fraudster's perspective to gain financially.

BIN attacks are difficult to prevent. Although BIN attacks represent fraudulent attempts, they rarely result in a loss to the merchant. The authorization attempts are never posted. These BIN attacks are merely a test to gain a range of compromised and validated card details that can then be used to perpetrate future higher value card not present (CNP) transactions.

However, there are ways to use a BIN attack to your advantage with minimal interference to a customer's experience.

What Is Your Defense?

Here are three things that you can do to stop a BIN attack:

1. A BIN attack that uses a bot is sometimes over within minutes. A 100% real-time fraud detection system is crucial to monitor low-value transactions. Using this capability makes it easy to profile small-value transactions on the same BIN, to the same merchant, and for the same amount within a short period of time.
2. When a bank observes a BIN attack, the bank can gather the cards that were exposed and mark them as an at-risk group in a hot list. Any transactions on the cards can then be assessed proactively to impose stricter measures for future online merchant purchases (for example, blocking all transactions from that channel for a certain period of time). Again, a sophisticated anti-fraud system that profiles all transaction records is required to enable such real-time analytics to be applied.
3. A sophisticated customer notification mechanism (for example, two-way SMS text messages) between the bank and card holders can help save legitimate customers from the impact of a BIN

attack. Ideally, the SMS text message requires a simple Yes or No response to confirm the validity of a suspicious transaction. If the customer does not respond within a certain time period, a time-out setting should be defined to block the card. This approach is much more effective than a single SMS text message notification that requests the customer contact the card center if the transaction is not authorized, leaving the transaction in an unknown state for a longer period of time.

BIN attacks, which have been used by fraudsters for a long time, will unfortunately continue. If banks staggered the sequencing when they issue cards, the consistency between sequential card numbers and the expiration date would be greatly reduced. Also, if merchants began to enforce additional security measures that are available, including demanding built-in card verification options and the use of 3D Secure 2.0, the opportunity to be defrauded would be reduced. Issuing banks must incorporate these mitigation steps for BIN attacks into their fraud detection systems.

Using SAS Fraud Management to Detect BIN Attacks

Overview

The following procedure shows how to set up a rule within SAS Fraud Management to protect your card base from BIN attacks.

Create a User Variable Segment

This rule tracks information about CNP transactions, specifically keyed and e-commerce point-of-sale entry mode transactions that occur for a merchant. You first need to create a user variable segment to uniquely identify the merchant. To ensure that the merchant is uniquely identified, select suitable key fields for the user variable segment. You can select multiple fields as long as they do not total more than 100 bytes.

If you do not choose a suitable user variable segment key, the system might experience cardinality contention due to volumes. *The locking of the user variable segment key record, which is often required to ensure the accuracy of data, prevents other transactions from using the same record. Only one transaction can update the row at a time, so other transactions must wait until the current transaction is finished. Although this takes milliseconds, a buildup of waiting transactions can occur. Cardinality issues occur when multiple transactions are trying to update the same user variable segment key within a short period of time.*

Note: The user variable segment prefix that is used in the following rule code is M. If you are already using the prefix M, you can choose another prefix, but the rule code needs to be updated to replace _m_ with the user variable prefix that you chose.

Remember to test new user variable segments for the effects of cardinality, should it occur.

Even without the impact of cardinality, the maturation of a user variable segment, once introduced, can have technical considerations that are not apparent to the bank. Consider the deployment of the user variable segment (even without a rule that references the new user variable segment).

Before implementation, consult with your internal IT department and SAS experts, including your Technical Support Account Manager (TSAM).

Create a User Variable

Now that you have created a user variable segment, you can create the user variables to use in the rule.

Note: You can use the replication count feature to create copies of the user variables.

Table 1. *Credit Card Numbers*

Name	Type	initial Value
_m_CM_PAN_1	Character 24	Not applicable
_m_CM_PAN_2	Character 24	Not applicable
_m_CM_PAN_3	Character 24	Not applicable
_m_CM_PAN_4	Character 24	Not applicable
_m_CM_PAN_5	Character 24	Not applicable
_m_CM_amount_1	Numeric	0
_m_CM_amount_2	Numeric	0
_m_CM_amount_3	Numeric	0
_m_CM_amount_4	Numeric	0
_m_CM_amount_5	Numeric	0
_m_CM_Country_code_1	Character 8	Not applicable
_m_CM_Country_code_2	Character 8	Not applicable
_m_CM_Country_code_3	Character 8	Not applicable
_m_CM_Country_code_4	Character 8	Not applicable
_m_CM_Country_code_5	Character 8	Not applicable
_m_CM_MCC_1	Character 8	Not applicable
_m_CM_MCC_2	Character 8	Not applicable

Name	Type	initial Value
_m_CM_MCC_3	Character 8	Not applicable
_m_CM_MCC_4	Character 8	Not applicable
_m_CM_MCC_5	Character 8	Not applicable
_m_CM_Tran_Expiry_1	Date	0
_m_CM_Tran_Expiry_2	Date	0
_m_CM_Tran_Expiry_3	Date	0
_m_CM_Tran_Expiry_4	Date	0
_m_CM_Tran_Expiry_5	Date	0
_m_CM_Date_Time_1	DateTime	0
_m_CM_Date_Time_2	DateTime	0
_m_CM_Date_Time_3	DateTime	0
_m_CM_Date_Time_4	DateTime	0
_m_CM_Date_Time_5	DateTime	0

Declare Variable Rule Arrays

Variable rule arrays are declared to group user variables in order to store the following information from the last five CNP keyed or e-commerce transactions that have occurred at the merchant's:

- card numbers (_m_CM_PAN_1 – _m_CM_PAN_5)
- transaction amounts (_m_CM_amount_1 through _m_CM_amount5)
- merchant country code (_m_CM_Country_code_1 through _m_CM_Country_code_5)
- merchant category code (MCC) (_m_CM_MCC_1 through _m_CM_MCC_5)
- expiry date of the transaction (_m_CM_Tran_Expiry_1 through _m_CM_Tran_Expiry_5)
- date and time of the transaction (_m_CM_Date_Time_1 through _m_CM_Date_Time_5)

Copy the following rule code into your variable rule:

```
if ucm_pos in ('00','01','81','8E','8F') and ucm_card_present = '1'
then do;

    %DECLAREARRAY(&rule.CM_PAN,_m_CM_PAN_1 - _m_CM_PAN_5);
    %DECLAREARRAY(&rule.CM_amount,_m_CM_amount_1 - _m_CM_amount_5);
    %DECLAREARRAY(&rule.CM_Country_code,_m_CM_Country_code_1
    %DECLAREARRAY(&rule.CM_MCC,_m_CM_MCC_1 - _m_CM_MCC_5);
    %DECLAREARRAY(&rule.CM_Tran_Expiry,_m_CM_Tran_Expiry_1 -
    _m_CM_Country_code_5);
    %DECLAREARRAY(&rule.CM_Date_Time,_m_CM_Date_Time_1 - _m_CM_Date_Time_5);

    %SHIFTHISTORYARRAY(&rule.CM_PAN);
    %SHIFTHISTORYARRAY(&rule.CM_Amount);
    %SHIFTHISTORYARRAY(&rule.CM_Country_Code);
    %SHIFTHISTORYARRAY(&rule.CM_MCC);
    %SHIFTHISTORYARRAY(&rule.CM_Tran_Expiry);
    %SHIFTHISTORYARRAY(&rule.CM_Date_Time);

    %SET(&rule.CM_PAN,1) = hqo_card_num;
    %SET(&rule.CM_Amount,1) = tca_client_amt;
    %SET(&rule.CM_Country_Code,1) = hct_term_cntry_code;
    %SET(&rule.CM_MCC,1) = hct_mer_mcc;
    %SET(&rule.CM_Tran_Expiry,1) = ucm_msg_exp_date;
    %SET(&rule.CM_Date_Time,1) = dhms(rqo_tran_date,0,0,rqo_tran_time);

end;
```

Create the Authorization Rule

The authorization rule checks whether there have been five CNP transactions in a row at a merchant, specifically keyed and e-commerce point-of-sale entry mode transactions. Each transaction is made with a different card, but the following conditions are met:

1. The cards have the same BIN (first 12 digits).
2. All of the transactions are for the same amount.
3. The merchant has the same country code on all the transactions.
4. The merchant has the same MCC on all the transactions.
5. The expiry date is the same on each transaction.
6. All of the transactions occur within one hour.

If the conditions are met, then the rule declines the transaction and creates an alert.

Certain high-volume e-commerce merchants that might cause unnecessary false positives by being in the rule can be excluded via a lookup list.

Copy the following rule code into your authorization rule.

When selecting the alert type, choose a merchant-level alert type, if available. This alert view enables your analyst to see all the transactions that occur at the merchant's, which helps the analyst identify whether it is a BIN attack.

```
if ucm_pos in ('00','01','81','8E','8F') and ucm_card_present = '1' and not
   %LISTCONTAINS (High_volume_merchants, hct_term_owner_name)
then do;

   if
   /* each transaction is made with a different card */
   (
   COMPRESS(hqo_card_num) ne (COMPRESS(_m_CM_PAN_2))
   or COMPRESS(hqo_card_num) ne (COMPRESS(_m_CM_PAN_3))
   or COMPRESS(hqo_card_num) ne (COMPRESS(_m_CM_PAN_4))
   or COMPRESS(hqo_card_num) ne (COMPRESS(_m_CM_PAN_5))
   )
   /* Card range is the same for all transactions */
   and substr(hqo_card_num,1,12) = SUBSTR(_m_CM_PAN_2,1,12)
   and substr(hqo_card_num,1,12) = SUBSTR(_m_CM_PAN_3,1,12)
   and substr(hqo_card_num,1,12) = SUBSTR(_m_CM_PAN_4,1,12)
   and substr(hqo_card_num,1,12) = SUBSTR(_m_CM_PAN_5,1,12)

   /* Amount is the same for all transactions */
   and tca_client_amt = _m_CM_Amount_2
   and tca_client_amt = _m_CM_Amount_3
   and tca_client_amt = _m_CM_Amount_4
   and tca_client_amt = _m_CM_Amount_5

   /* MCC is the same for all transactions */
   and hct_mer_mcc = _m_CM_MCC_2
   and hct_mer_mcc = _m_CM_MCC_3
   and hct_mer_mcc = _m_CM_MCC_4
   and hct_mer_mcc = _m_CM_MCC_5

   /* Expiry Date is the same for all transactions*/
   and ucm_msg_exp_date = _m_CM_Tran_Expiry_2
   and ucm_msg_exp_date = _m_CM_Tran_Expiry_3
   and ucm_msg_exp_date = _m_CM_Tran_Expiry_4
   and ucm_msg_exp_date = _m_CM_Tran_Expiry_5


   /* DateTime is within 1 hour */
   and dhms(rqo_tran_date,0,0,rqo_tran_time)
   - _m_CM_Date_Time_5 <= dhms(0,1,0,0)

   then do;
   %ACTION_DECLINE;
   %ACTION_ALERT;
   end;
end;
```

After you have created both rules, follow your normal procedures for testing and promoting the rules.

For more advice and support about preventing BIN attacks and implementing strategies, contact your local or global fraud expert at SAS.

Resources

SAS Fraud Management documentation is intended for use by existing customers and requires an access key. You can obtain the access key from your SAS consultant or by contacting SAS Technical Support. To expedite your request, please include **SAS Fraud Management** in the subject field of the form. Be sure to provide the SAS Site Number for your software license along with your request.

Release Information Content Version: 1.0 May 2021