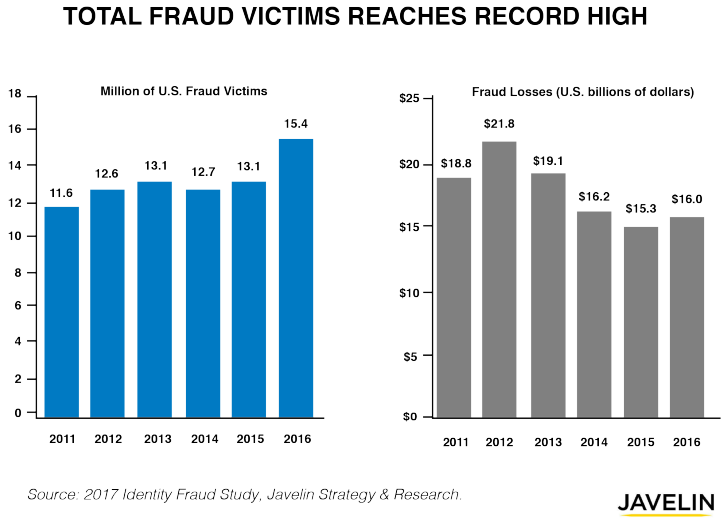
AUTHORITI™

Flipping the Model

A Disruptive Model
for Authorizing Transactions

Identity theft is not only on the rise, it's rising fast. The problem affected an estimated 15.4 million US consumers in 2016 and is currently growing at 18% year over year.[1]



Many of the incidents we see today and will see tomorrow stem from data breaches in which Social Security Numbers (and other Personally Identifiable Information, or "PII") are leaked or stolen. Fraudsters then use this information to establish credit in the victim's name, tap existing credit, improperly claim tax refunds, commit insurance fraud, etc. A huge percentage of consumer complaints to the Federal Trade Commission are fraud related.[2]

Why is this happening? The root cause of the problem is the usability of stolen PII in today's transaction model. To examine this, let's first eliminate the confusion by defining three core concepts: Identification, Authentication, and Authorization.

- ***Identification* is any way to uniquely distinguish yourself from others. It's how you claim that you are you.** The most common national identification (ID) number in the US is a government-issued Social Security Number, but there other are many other IDs as well, including passport numbers, driver license numbers, bank account numbers, phone numbers, credit card numbers, and so on. They don't have to be numbers; login names online and email addresses are also identifiers. Even your name is a form of identifier – but not a good one, because others might legitimately have the same name.

- ***Authentication* is how you prove that you are you.** Authentication can be done though a secret password or receiving a code that shows you can access a device that only you should possess (such as your phone). Authentication can also use behavioral and biometric techniques like the machine you login from, scanning your fingerprints, camera-based facial recognition, and others.

- **Once authenticated, *authorization* sets (or restricts) the transactions that are permitted.** A good example of an authorization is the maximum daily withdrawal amount from an ATM.

[1] Javelin Strategy and Research (2017)

[2] Federal Trade Commission (2017)

Identification information is meant to be shared. Authentication and authorization information is not.

Failure of the Current Model

As we've seen, this model has failed because the consumer does not have control over his or her own transactions, and financial institutions do not have confidence in the transactions.

Imagine if your bank created an online account for you and used your email as your ID (identifier). Now imagine if, for simplicity, they gave you a default password that was also your email address. You probably wouldn't be very comfortable with that, because a lot of people know your email address and could therefore access your accounts.

Remember, identification information is meant to be shared, authentication information is not. Combining them is a bad idea.

Which brings us to well-known IDs such as Social Security Numbers. Although an SSN is, by definition, an identifier, early Social Security cards came stamped with the words "not to be used for identification." This was a clear sign of the confusion ahead. What they meant was that possessing the card should not be used to prove one's identity – that's authentication. Because, of course, someone could have stolen a wallet containing the card.

Over time, the fact that everyone had a unique government ID number made it convenient to broadly use SSNs as ID. The fact that the number wasn't publicly published made it convenient to assume is was also a secret – so, knowing the number became like knowing a password. However, as everyone knows, it's a secret that each of us has been asked to share with a lot of people over the years. And the first rule of secrets – like passwords – is to never share them.

The more we share our SSNs with banks, motor vehicle departments, tax authorities, employers, insurance companies, brokerages, doctors' offices, utilities, and credit agencies, the more opportunities there are for the secret to leak out.

Remember the example of online banking above, where we explained why a widely shared identifier (email) shouldn't be used as a secret password? That's exactly what we have done with Social Security Numbers. In fact, today, we are repeating the mistake in more places. For example, as long as someone knows your credit and debit card numbers, they can spend your money.

What We Really Care About Is Misuse

Social Security Numbers have one more feature that makes them bad secrets. They are hard to change once they do get out. They stay around forever. (The same issue is true of biometric authentication on a smartphone. Ever try to change your fingerprint?)

Consumers want their information protected, of course, but what they really care about is that it is

not misused. Over-focus on hiding identifiers such as SSNs has happened at the expense of ensuring that they are not used fraudulently.

The misuse of identifiers as secrets isn't limited to Social Security Numbers:

- Credit and debit card numbers (with an accompanying fixed PIN) also need to be kept secret to combat fraud. Yet we freely share them with waiters, store clerks, and online shopping sites.
- As long as you have the answer to "out of wallet" questions such as your mother's maiden name, account numbers can be used to reset online banking passwords. The answers to these "secret" questions can be easily learned through social media and data breaches.
- Knowing an online ID and password allows fraudsters to transfer money out of a bank or brokerage account. In fact, consumers have many "secret" identifiers, sometimes combined with other "secret" information. If leaked, these secrets can be used to take over accounts and commit fraud.

Solving the Wrong Problem

This confusion has resulted in significant efforts to solve the wrong problem. We aren't able to keep shared identifiers secret for authentication, and we certainly shouldn't use them for authorization.

Even the use of one-time PINs, such as Google authenticator or online banking authentication, confuses authentication with authorization. Additionally, these authentication PINs are subject to being intercepted or redirected. By hacking accounts at a wireless carrier, fraudsters can get the text messages containing PINs without the knowledge or authorization of their owners.

Hacking the systems that generate or store PINs is another way to compromise authentication. Some companies even ask potential fraudsters to provide the phone numbers to which new PINs should be texted. This completely invalidates the channel as a means of secondary authentication. Authentication is weakened and authorization is lost when a company needs user authorization to proceed – and it's the company, not the user, who generates a PIN.

Today, if an institution wants user permission, they generate a permission PIN that they send to the user … to send back to them! And we wonder why we still have fraud.

Today's model is upside down:

- Consumers don't want the financial institution or a remote service to authorize their transactions – they want the ability to authorize themselves.

- Fraud departments don't want to be in the no-win position of having to guess if a transaction is legitimate. Being too permissive results in losses and being too restrictive results in client and business dissatisfaction.

Flipping the Model

At Authoriti, we are completely flipping the model. Through the use of an Authoriti Permission Code™, we enable consumers to tell institutions which transactions are authorized and which ones aren't. The Authoriti Permission Code model is based on the following premises:

- Authorization is clearly distinguished from authentication, and the need to keep identifiers a secret is eliminated.
- Authorization comes from the consumer and not the institution.
- This consumer authorization contains limitations on how, when and where the Permission Code can be use – eliminating the opportunity for misuse.
- The need for a central repository where passwords and usable PINs are kept is also eliminated.
- The need for an outbound channel that is susceptible to being intercepted or redirected is eliminated.

With the Authoriti Permission Code, consumers, not the firm, generate one-time PINs to authorize transactions. A usable Permission Code is not stored anywhere; it's signed with a private key and verified with a public key. But the real magic is that the Permission Code is cryptographically bound to both the ID and the specific transaction. It has embedded within it restrictions that prevent it from being misused, even if intercepted.

If a consumer authorizes an immediate wire transfer out of a specific account in a NY bank branch, the Permission Code can't be used for any other account, purpose, location, or time. Permission Codes are small enough to be client-friendly, but pack lots of information.

Validating a Permission Code takes a simple call to the RESTful Authoriti web service. No back-office changes to databases are needed; account numbers are still the identifier of choice and the Permission Code can be safely discarded once checked.

In the Authoriti model, the consumer is able to control the use of his or her information instead of worrying about keeping it a secret. Institutions that process or share information gain confidence that the real owner authorized its use for the intended purpose. The chances of fraud and misuse are dramatically reduced. Rather than merely shifting fraud risk, our model seeks to eliminate risk completely.

About the Authoriti Network

The Authoriti Network was founded in 2017 to identify new approaches to prevent misuse of Identifiers and Personally Identifiable Information. Our founders have significant leadership experience dealing with InfoSec at-scale at the world's leading financial institutions. Authoriti develops the Authoriti Permission Code™, which puts control of transactions in the hands of the consumer and gives institutions the confidence that the transactions are authorized. Please visit us at Authoriti.Net