

THE NEEDLE IN THE HAYSTACK

Digital channels make life simpler, but they also make detecting cyberattacks as difficult as finding a needle in a haystack. Fortunately, machine learning optimizes security across various levels, making such a task that much more achievable.

By Hartmut Keil and Aldo Rodenhäuser

Using machine learning (ML) it is possible to find the needle in the haystack – often before it pricks you. In the cyber security environment, the needle represents a nonlegitimate event such as the initiation of a transaction by an assailant. The haystack represents the millions of legitimate events, whether they involve a user accessing a system online or they are generated from internal processes. Machine learning has the potential to redefine the balance between an assailant and a defender.

MACHINE LEARNING HAS THE POTENTIAL TO REDEFINE THE BALANCE BETWEEN ASSAILANT AND DEFENDER.

Imagine the following situation: A customer phones his bank and reports a transaction that he supposedly did not execute himself. Said transaction took place on a digital channel.

This is where the work of the digital forensic scientist starts. The first thing the scientist does is to collect all available and relevant information, which is usually a manual and often lengthy process that involves asking of questions such as:

- At what time and from which place did the customer/assailant use the digital channel?
- Which device did the customer/assailant use? Does the communication pattern of the device used correspond to known patterns?
- Does the sequence of queries correspond to a legitimate, established pattern? Is it typical for this user?
- At what time and from which place did the customer issue the queries? Who were the recipients and what were the amounts to date?

In most cases, only some of this information is available. Either because it was not recorded by the systems or because it

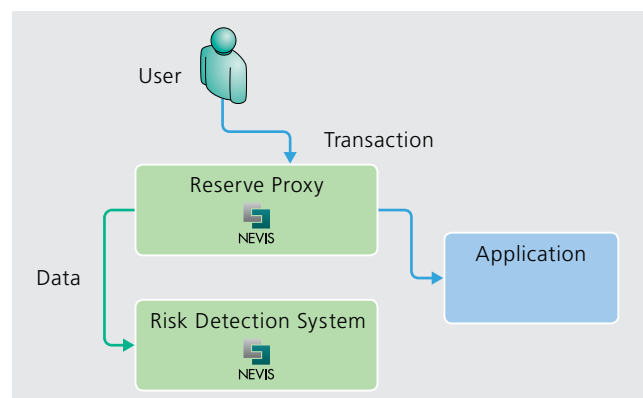
was recorded but has since been deleted. We are, after all, talking about enormous quantities of data here. And on the application side, it was perhaps not clear during development, which data would be relevant and thus worthy of recording by the system.

The digital forensic scientist's second step is to correlate the data collected from different systems and communication layers with one another and extract a pattern. This takes place manually in most cases.

In a last step, the forensic scientist is now capable of determining whether the transaction in question was executed by the legitimate user or whether he/she is indeed dealing with an attack and, if so, what the nature of the attack is.

The aim of the analysis is essentially to estimate to what degree of probability the transaction corresponds to a previously observed pattern.

It would seem logical to build a system to automate this process. The system should detect in real time whether a transaction is initiated by a legitimate user or not. This would have two advantages: It would reduce the expensive analyses and, at the same time, increase security.



User transaction protected by risk detection system.

Risk detection system replaces manual analysis

The system to be built is called a risk detection system. The system's first task is to collect data. If a reverse proxy is used, it can send all requests and the corresponding data (TCP/IP messages, SSL records, etc.) to the risk detection system.

The surrounding system landscape is not affected and is transparent for the application. The next steps of pattern extraction and the detection of nonlegitimate transactions take place in the risk detection system.

The effectiveness of such a system depends very much on the information available to it. The following examples illustrate this:

- The transaction in question is run by malware on the customer device. This can be detected only because of anomalies in the connection establishment.
- The transaction is executed by a member of the customer's

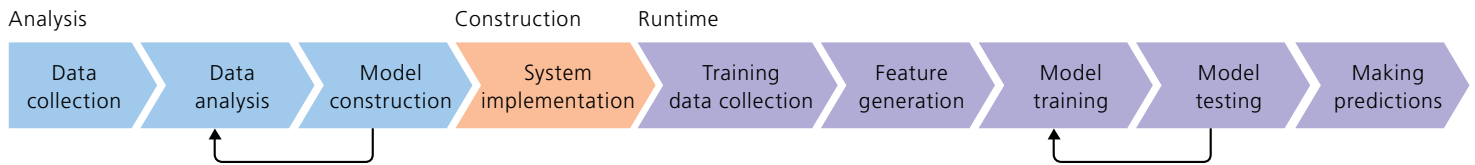
family after the person has authenticated him-/herself with the valid access data. This case can only be detected with biometric information.

The data generally available can roughly be divided into the following categories:

- Device and network information: all information that is influenced by the user's device. In the case of a web application this includes, for example, the browser used, the operating system or the hardware model.
- Context information: the information determined by the physical context such as location data (geo location) and time.
- Biometric information: all information which is determined solely by the user and/or his/her behavior, for example the way the user uses the computer keyboard or mouse.
- Applicational information: transaction data and patterns.



Aldo Rodenhäuser and Hartmut Keil: Their main focus is on security.



The individual steps of risk detection.

Defining ML aims

We now want to describe the job of our risk detection system in the context of ML in more detail.

First of all it collects data, referred to below as training data. Then, or parallel to that, it generates features from this "raw" data. These features are used as input for the model. Secondly, the model is trained, something we referred to above as pattern extraction. A model usually contains two aspects: certain domain-specific statements which the model has to infer and static assumptions about the data or features which have to be fulfilled.

**THE MODEL COLLECTS
DATA AND GENERATES FEATURES
FROM THE DATA SO THAT
THE MODEL CAN BE TRAINED.**

This training phase basically consists of optimizing certain features of the model so that it concurs as best as possible with the training data.

The last step is to use the model to decide whether a transaction is legitimate or not. This forecast is referred to as a prediction. At this point we have to define in more detail which prediction our risk detection system or its model should make. Our model should decide whether a transaction is legitimate, in other words whether it corresponds to the pattern of legitimate transactions, or whether a transaction is not legitimate, in other words does not correspond to the pattern of legitimate transactions.

This specification is necessary if we want to phrase the task in ML terminology. A distinction is made between the following tasks in ML:

- The prediction of whether a transaction has the label "legitimate" or «not legitimate» is referred to as the classification.
- The prediction of whether a transaction is similar to the known transactions of the training data or not is referred to as novelty detection.

How can this distinction be explained and how can it be motivated? In classification it is the model's task to detect whether a fruit is an apple or a pear. The training data has around the same number of apples as it does pears. In novelty detection

it is the model's task to detect whether a fruit is an apple or not. The training data contains only apples.

With this background we can now discuss the basic task of our risk detection system or its model: We know that in the haystack, i.e. the training data, there are virtually no needles or that we cannot detect the ones that are there. This is why the model's task has to be novelty detection. What is important now is that our model can deal with this situation. It has to be capable of minimizing the influence of the outliers (needles) in the training data to the prediction (this characteristic is an example of the above-mentioned statistical assumptions of our model).

The features used and the model are the central aspect of our system: Which features are to be generated from the data and which domain-specific and statistical conditions must our model fulfill? These questions should be clarified in an analysis phase, generally using data which has already been collected.

**IN EVERY WEB-BASED
APPLICATION THE USER'S IP
ADDRESS AND THE TIME OF
TRANSACTION ARE KNOWN.**

Example of context information

Context information is part of the data which is generally available. For example, in every web-based application the IP address of the user and the time of a transaction are known.

The IP address as such provides only very little information. Using special geo location databases, which usually have to be paid for, it is possible to generate features from the IP address, such as country, region, city, geographic degree of latitude and longitude.

When it comes to the time, you have to decide on one of two possibilities: the local time of the sender of the transaction (user) or the local time of the recipient (server on which the application runs). Other features can be generated from the time, such as second, minute, hour, day, month and year.

Which time and geo location features are used depends on the specific case. The following example shows how domain knowledge influences the choice of features and the model: Let's take an application which sales reps of a multinational company use solely for business purposes. The sales reps also

work in places which are not registered completely in the geo location database used. In this case it is obvious to use just the country as geo location feature. For the time features, day and hour of the local time of the user are sensible choices. Furthermore we know that a user always uses the application at a similar time. The selected model must imply this domain knowledge, i.e. the independence of time and geo location features.

What happens now if our model does not imply this assumption? It would possibly learn from the training data that a user in a specific country does not use the application on certain days. This, however, is only a result of the quantity of training data not being sufficiently large.

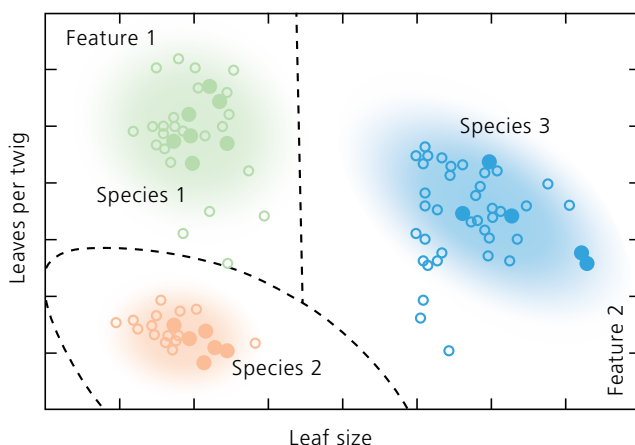
And what happens if we use other features? Let's take the city as a further geo location feature. We know that the city is not contained in the geo location databases for all IP addresses. For the model this means that it also has to be able to deal with incomplete data – which limits the quantity of suitable models.

Our risk detection system can now use the selected model to decide whether a transaction is legitimate or not. A user's transaction is legitimate if the time and geo location features correspond to the learned pattern. If this is not the case, the transaction is not legitimate.

THE IDEA OF IDENTIFYING A BROWSER UNIQUELY IS NOT NEW.

Example of device and network information

Like context information, device and network information are generally available. The idea of identifying the browser uniquely is not new. Using the JavaScript of new HTML5 functionalities, it is possible to glean more and more information from the browser and thus be able to identify it virtually



Browser identification via features.

uniquely. The disadvantage: This approach only works if the user's device is a browser.

A more general possibility of identifying the browser or the user's device is to learn certain patterns in the structure of the TCP/IP and SSL connection. This approach is based on the fact that parts of communication protocols such as TCP/IP and SSL allow a certain degree of freedom in implementation. This is how it is possible, for example, to distinguish two SSL implementations using the session ID. The specification of SSL requires merely a unique session ID of a maximum length. The appearance of this session ID is a detail of the individual implementation.

IN E-BANKING, THE MODEL LEARNS THE PATTERN OF THE CONNECTION ESTABLISHMENT.

As with the context information, you have to find out which features are to be generated from the TCP/IP and SSL data packages. This requires precise knowledge of the protocols.

But unlike the case of context information, there is, however, virtually no other domain knowledge here to find a model. This situation reveals the power of ML: Over the last decades, very general and flexible models have been constructed. One category of models (support vector machines) has been successfully implemented for the classification of devices.

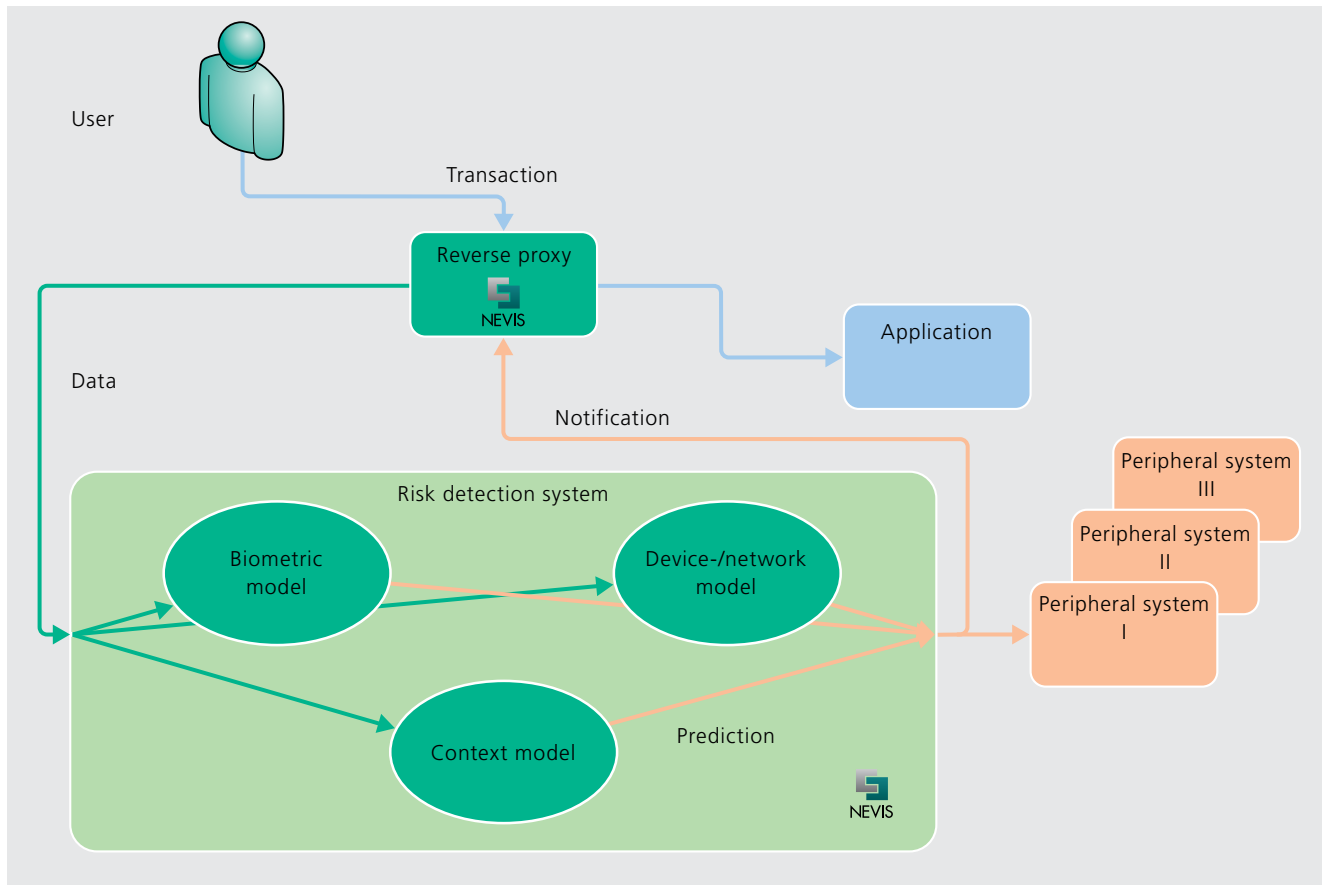
An unknown data point is now classified by the decision boundary: The label is predicted according to the side of the decision boundary it lies on.

Support vector machine models are suitable not only for classification but also for novelty detection. A further important aspect: They fulfill our requirement to be able to deal with outliers (needles) in the training data (haystack). This type of model is referred to as soft-margin support vector machine.

There are several possible approaches of how our risk detection system can use this identification of the user device. Take a look at the following two examples:

The application is the order system of a stock exchange. "Users" are the systems of the associated banks. In this situation it is sensible for our model to learn the pattern of how each individual system connected establishes the TCP/IP and SSL connection. A transaction is then recognized as not being legitimate when the connection establishment does not correspond to this pattern.

In a bank's e-banking, the model learns the pattern of the TCP/IP and SSL connection establishment of all browsers used. It is not the case of one model per user being trained as in the first example. A transaction that has been initiated by malware, for example, is detected as not legitimate as the connection



Architecture of a risk detection system.

establishment does not correspond to any of the browsers used. Here, the approach used in the first example would also be possible but the massive number of users would make the quantity of training data to be saved immense.

Putting it all together

Using examples, we have shown how ML can be applied to learn whether a transaction is legitimate or not from context and device/network information. ML approaches have also already been established for biometric and applicational information.

For our risk detection system to fulfill its task perfectly and provide protection for as many types of application as possible, it seems only natural to combine all these approaches. A central risk detection system thus always has an integrative character: It provides information for the different approaches, prevents attacks at the early stage on the reverse proxy and notifies peripheral systems.

This makes full use of all the available information to detect attacks. For the assailant, the complexity and required knowledge thus increase incredibly: He not only has to steal access information from his victim, he also has to look like the victim, behave like the victim and be at the same places at the same times. ■

Hartmut Keil

Hartmut Keil, MSc in Physics/CAS ETH Visual Computing, joined AdNovum in 2000 as a software engineer. For several years he was involved in the development of middleware products for e-banking and e-government applications. Currently, he focuses on the use of machine learning in the area of security. In his free time, the nondigital world is on the agenda, where he builds or repairs things with his children.

Aldo Rodenhäuser

Aldo Rodenhäuser is Head of Security Consulting at AdNovum and has more than ten years of experience as an IT security analyst and advisor. In his work, he focuses on cyber security, identity and access management as well as mobile security. On a regular basis he provides advice to leading global banks and government institutions in Asia and Europe on their security strategies and architectures as well as their organizational and technological risks. His private program includes expeditions to foreign countries, where he enjoys nature and culinary highlights.