Technical Brief

API-First Security

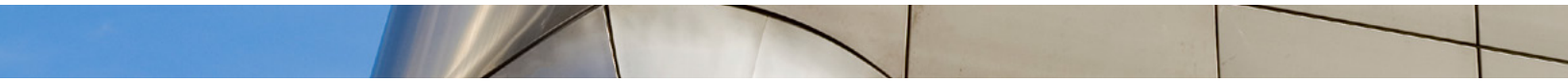
Don’t Build Your Own Maginot Line

apigee

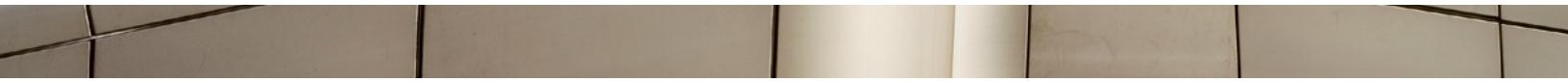
API-First Security

apigee

Table of Contents



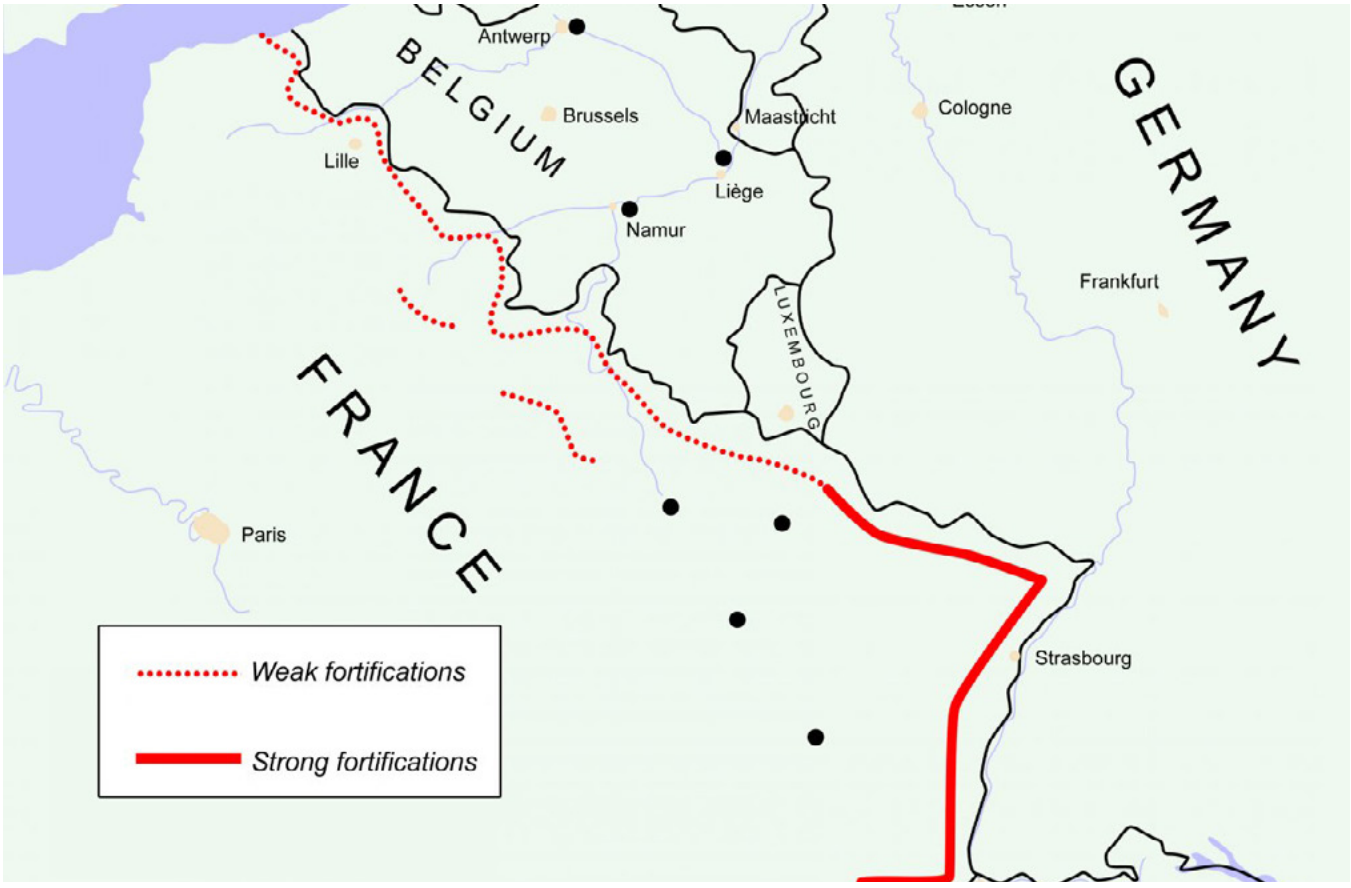
Introduction: The Maginot Line



The Maginot Line is remembered as one of history's most costly strategic blunders. France's massively complex, heavily fortified, and expensive 940-mile line was built before WWII to keep Germany at bay. But France's foes simply skirted the fortification via Belgium, and conquered France in six weeks.

The Maginot Line failed because it wasn't designed to counter the newer threats and tactics employed by Germans at that time; it wasn't flexible enough to adapt to new, nimbler threats.

Digital security strategies can exhibit a similar weakness, as they're not designed to adapt to new and unexpected threats (the Heartbleed vulnerability is a good example).



Finding the Right Security

Omnichannel security

A typical enterprise has hundreds of business applications hosted in private or public clouds that interact with their users (customers, partners, and employees) spread across geographies and time zones. These interactions take place via a variety of channels: web, mobile, APIs, VPNs, cloud services, and sometimes via contactless payment terminals supporting Apple Pay.

As a consequence, enterprises need to plan for an omnichannel user engagement model that provides a consistent user experience, with security and privacy built into all channels. And it should be supported by a security architecture that's responsive enough to enable enterprises to prevent, detect, and react to newer threats, in near-real time.

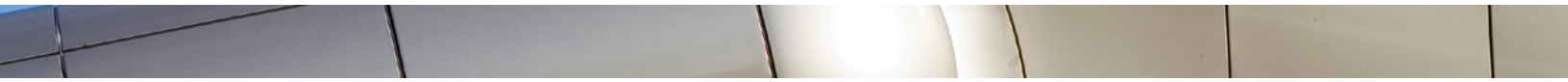
The problem with bolt-on security

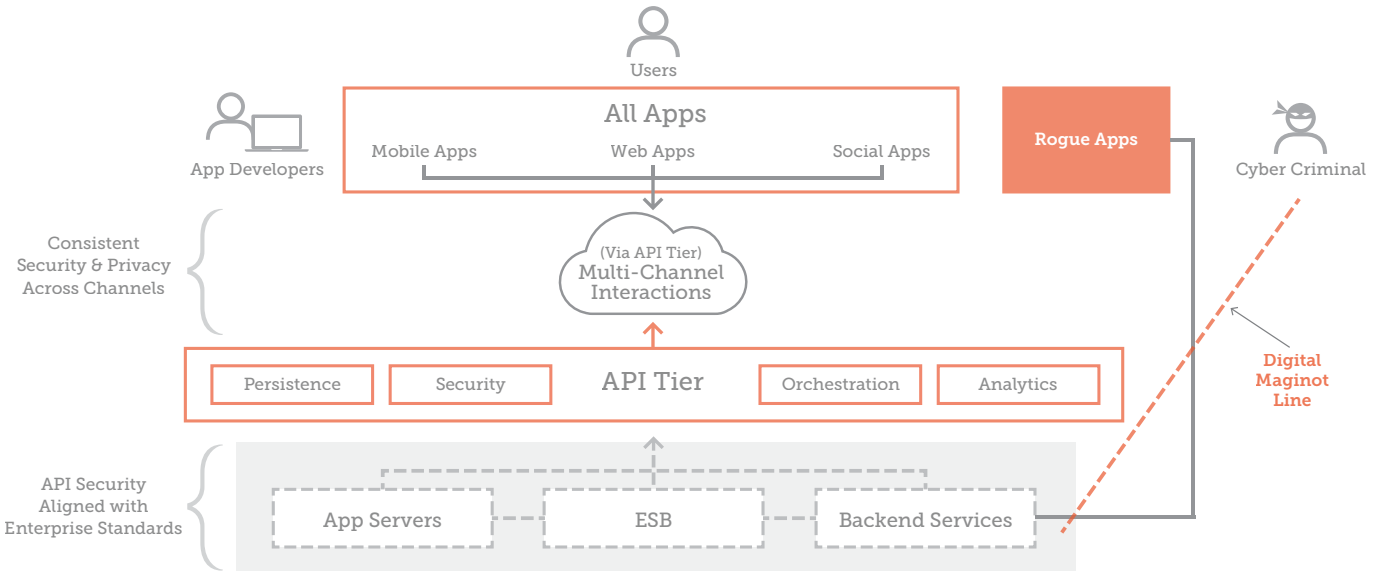
One way to achieve this outcome is to secure each channel independently using point solutions. This approach usually involves a bolt-on security model, in which the protection mechanism addresses specific threats for individual channels—securing mobile apps using a custom authentication without OAuth, SSO (Single Sign-on), and support for federated identity store, for example.

With this approach, a business won't be able to support new security requirements as user engagement moves from a single channel to multi-channel mode. And, from the perspective of the chief information security officer, a channel-specific security approach may provide the Maginot Line-like perception of strong fortifications, but won't be able to withstand new threats that blend across channels. Ultimately, the channel-specific security strategy leads to security weakness, with inconsistent user experiences across channels.

As a consequence, enterprises need to plan for an omnichannel user engagement model that provides a consistent user experience, with security and privacy built into all channels. And it should be supported by a security architecture that's responsive enough to enable enterprises to prevent, detect, and react to newer threats, in near-real time.

API-First Architecture





An effective digital security strategy and architecture that adapts quickly to new threats can be realized by building on an API-first architecture. In this model, every mobile, web, and IoT device interacts with business applications via APIs. An API-first architecture facilitates interactions within and across channels via an API tier that's distributed to scale with user engagement. By building security into the API tier, an enterprise can enforce a consistent protection mechanism across all channels.

One of the advantages of API-first architecture is the ability to expose APIs as "products" with built-in authentication, authorization, and threat protection. API products enable the right level of security for users, irrespective of the channel through which the interaction takes place.

For example, a "private" API product can expose private APIs that support SSO for employees using a corporate active directory, while a "public" API product supports SSO using a federated or social identity. As a result, you can manage security across channels in a very consistent way while preserving the agility of the business units exposing different sets of APIs.

API-First Security Principles

Every enterprise should follow API-centric security principles, as they are a critical part of an API-first architecture:

Expose your APIs as API products with fine granular access control built in. Your mobile app designed for users of your public API product can support social logins, such as Facebook. A mobile app designed for your employees using a private API product may use your corporate directory for authentication and authorization needs Both products should support OAuth and authorization scope to limit access to different sets of APIs.

Always configure security; code only when absolutely necessary. Using the API product approach, you can configure security policies that support authentication, authorization, and threat protection that persist across channels. Security configuration provides a better experience for developers and helps them adapt security features faster. Threat protection features such as rate limiting and OWASP Top-10 Injection protection act as a defense layer for all channel interactions. But you'll still need the option to code security for cases where configuration may not be an option. For example, you may need to query your internal SIEM (security information event management) system for malicious sets of IP addresses (black lists) and block the threats across all channels.

Log all API interactions across channels to get an end-to-end view of user, device and app activities. This approach helps provide correlations of user activities across all channels and identify any abnormal behavior that deserves investigation. API analytics can provide insights into traffic spikes from certain devices, apps, IP addresses, or geographies.

Always layer API security with other controls for a defense-in-depth strategy. Be sure to employ additional access protection between apps and API services. An example: using IP-based whitelisting and two-way SSL/TLS in addition to OAuth-based security.

Remember, build your apps and services upon an API-first architecture that's resilient and adaptive to new threats. An API-first architecture enables consistent security policy enforcement across channels while improving user engagement and developer productivity.

With an API-first architecture approach, you can avoid the digital Maginot Line, where hackers exploit the weakest channel and create a costly data breach response situation for your organization.

apigee

About Apigee

Apigee delivers an intelligent API platform to accelerate the pace of digital business. We help companies – from disruptive start-ups to the Fortune 100 – use their enterprise data and services to create connected digital experiences for customers, partners, and employees. This is digital business.

For more information, visit apigee.com.

Where to go from here

For more on how to build a secure, end-to-end API architecture to support the digital enterprise, download the free eBook, "Securing the Digital Enterprise: Infrastructure Security for the CSO."

The Apigee Edge pplatform helps developers and companies of every size manage, secure, scale, and analyze their APIs. Get started with the right Apigee Edge today.

For additional resources, visit apigee.com/about/resources/

Share this eBook