Evolution of the Zero Trust Model for Cyber Security

Dr. Edward Amoroso
eamoroso@tag-cyber.com
Mob: (201) 454 - 1854

Thirty years ago, a researcher wrote a small piece of software that saved his company from a cyber attack by a college student. The software examined inbound TCP packets for ACK=0, which denoted the start of a session. If a new session looked funny for any reason, then the packet was dropped. Weird source IP addresses or unauthorized destination services were typical justifications for disallowing inbound packets. The code was well-written and it worked.

The code author was named Bill Cheswick and the company saved was Bell Laboratories. The college student was Bob Morris Jr. and his attack was the Internet Worm of 1988. (In an ironic twist, Bob Morris Jr. had a famous father of the same name who'd helped invent Unix at Bell Laboratories). And as for the small piece of software written by Bill Cheswick – well, it was arguably the first working enterprise firewall. (And no, the company did not patent the idea.)

Other organizations not running an early firewall were infected with the Morris worm because they assumed a so-called *full trust model* between entities on the Internet. Remember that when the Internet was invented, the presumption was that this new forum was to be used solely for trusted communication and sharing. Basic cooperative norms existed across the early Internet, and were respected by individuals, groups, companies, and other entities.

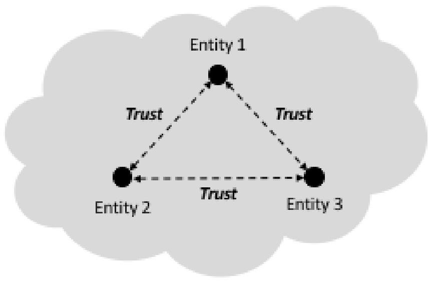

Figure 1. Full Trust Model

But by the mid-1980's, it was apparent that the optional norms and procedural controls on the Internet – usually to be followed voluntarily by network operators and systems administrators – were insufficient to secure the growing number of critical functions being hosted on this new global infrastructure. Individual and group-hacking soon emerged as a serious menace – and the Morris worm exemplified the cascading power of well-defined attacks.

Such security issues ushered in an era of *perimeter networking*. That is, soon every organization in the world created a firewall-based bunker, which provided a *partial trust model* between entities inside and outside an enterprise. Thus, if an entity was located inside the firewall, then it could trust any other entities within the perimeter – users, desktops, servers, printers, applications, and so on. If an entity was outside, however, then it was not to be trusted.

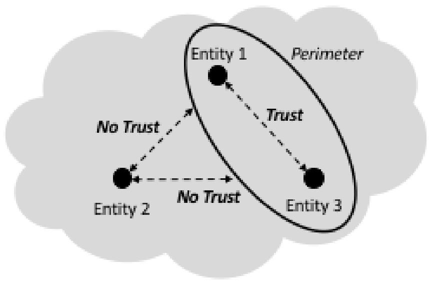

Figure 2. Partial Trust Model

This partial trust model has driven enterprise security for three decades. It's severe drawbacks, however, soon became evident. First, an entity could be inside the firewall, but also be malware-laden or disgruntled. Second, if an entity wanted to connect to the world, then the perimeter needed to allow email, remote access, and web – all open doors for hacking. And third, any mobile-connected device could wirelessly bypass any enterprise firewall.

To address these weaknesses, enterprise security teams completely redesigned the original perimeter concept, opting for a new model that combines the great modern themes of today – namely: *cloud*, *mobility*, and *apps*. This new model is based on the idea of mobile devices accessing cloud-hosted apps without the need for a perimeter. The resulting device-to-cloud operation produced something now referred to as a *zero trust model*.

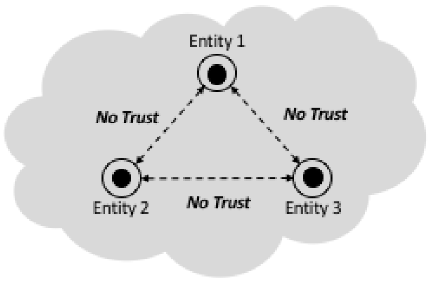

Figure 3. Zero Trust Model

To implement zero trust cyber security, represented by the little circles around each entity in figure 3, four components are required: *Devices*, *transport*, *cloud*, and *apps*. Let's examine the role of each in the context of a typical, modern enterprise deployment:

Devices. The security obligations for device stewards in a zero trust environment involve ensuring strong authentication to the device, protecting the mobile device operating system and application environment from malware, and establishing a secure launch point for connectivity to cloud-hosted apps.

Transport. The security obligations for transport infrastructure providers in a zero trust environment involve maintaining highly available communications, since device-to-cloud services are the new backbone for business and personal use, and supporting flexible means for protecting networks from attacks such as DDOS.

Cloud. The security obligations for cloud hosting providers[1] in a zero trust environment involve providing micro-segmentation for workloads, as well as supporting security and regulatory compliance demands imposed on any external hosting of critical enterprise applications. The cloud must also support secure device connectivity and network access control.

Apps. The security obligations for applications in a zero trust model environment involve supporting highly secure code, integrating security connectors to provide telemetry to cloud-hosted tools such as SIEMs, supporting secure device connectivity, and implementing proper access management and data security for any stored sensitive information.

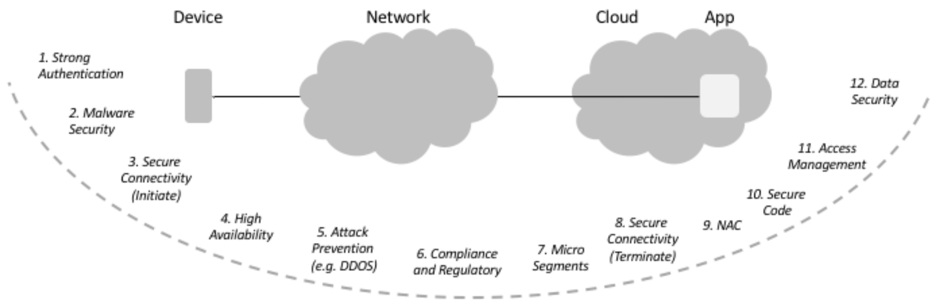

Figure 4. Security Controls for Device-to-Cloud in a Zero Trust Environment

An excellent example of a zero trust implementation is the Google BeyondCorp concept used for enterprise networking at the company.[2] The goal in Google's secure enterprise design is to remove dependency on its perimeter. Instead, a device-to-enterprise scheme uses company-provisioned device access to Google network infrastructure for secure access to required business apps. Below are the salient aspects of this design:

[1] Zero trust connectivity to a corporate enterprise results in the traditional "enterprise LAN" being viewed as a remotely accessible cloud. In this way, device-to-cloud access for a corporate app is no different conceptually than device-to-cloud access to Facebook. The difference is the access management at the cloud or app destination.

[2] See "BeyondCorp: A New Approach to Enterprise Security," by Rory Ward, and Betsy Byer. Presented in ;login, December 2014. https://storage.googleapis.com/pub-tools-public-publication-data/pdf/43231.pdf

Access Based on Credentials. In BeyondCorp, direct access is only permitted for devices that are procured and managed by Google. Device inventory is tracked, and configuration changes to devices are maintained in a meta-inventory. Certificates are issued and stored in trusted platform module (TPM) hardware on devices, and are referenced in an inventory database. A qualification process determines if an endpoint device is secure enough for access.

Access and Encryption. Access management is the central function for the BeyondCorp design, and is implemented based on the attributes of the requesting user, any relevant group memberships, and applicable HR-related job functions. Single sign-on (SSO) is supported using sort-lived tokens generated for use during sessions. All sessions originated at a provisioned device are also encrypted.

Untrusted Network. Google runs an unprivileged network for device access that is essentially public, but that is located in private Google address space. Only minimal external connectivity is included on this network to DNS, NTP, DHCP, and Puppet. RADIUS used to provide network access control (NAC) coordination to the desired VLAN for access to the Google application being requested.

Identical Local and Remote User Experience. The operational result of BeyondCorp is that the session experience for users accessing Google apps locally is essentially the same as for users accessing the same apps remotely. In this way, BeyondCorp elegantly removes the need for an enterprise perimeter, which implies all the security and operational benefits of a zero trust computing environment.