Changes to Trusted Certificate Authorities in Android Nougat

Posted by Chad Brubaker, Android Security team

In Android Nougat, we've changed how Android handles trusted certificate authorities (CAs) to provide safer defaults for secure app traffic. Most apps and users should not be affected by these changes or need to take any action. The changes include:

- Safe and easy APIs to trust custom CAs.
- Apps that target API Level 24 and above no longer trust user or admin-added CAs for secure connections, by default.
- All devices running Android Nougat offer the same standardized set of system CAs—no device-specific customizations.

For more details on these changes and what to do if you're affected by them, read on.

Safe and easy APIs

Apps have always been able customize which certificate authorities they trust. However, we saw apps making mistakes due to the complexities of the Java TLS APIs. To address this we improved the APIs for customizing trust.

User-added CAs

Protection of all application data is a key goal of the Android application sandbox. Android Nougat changes how applications interact with user- and admin-supplied CAs. By default, apps that target API level 24 will—by design—not honor such CAs unless the app explicitly opts in. This safe-by-default setting reduces application attack surface and encourages consistent handling of network and file-based application data.

Customizing trusted CAs

Customizing the CAs your app trusts on Android Nougat is easy using the Network Security Config. Trust can be specified across the whole app or only for connections to certain domains, as needed. Below are some examples for trusting a custom or user-added CA, in addition to the system CAs. For more examples and details, see the full documentation.

Trusting custom CAs for debugging

To allow your app to trust custom CAs only for local debugging, include something like this in your Network Security Config. The CAs will only be trusted while your app is marked as debuggable.

```
<network-security-config>
      <debug-overrides>
           <trust-anchors>
                <!-- Trust user added CAs while debuggable only -->
                <certificates src="user" />
           </trust-anchors>
      </domain-config>
 </network-security-config>
```

Trusting custom CAs for a domain

To allow your app to trust custom CAs for a specific domain, include something like this in your Network Security Config.

```
<network-security-config>
      <domain-config>
           <domain includeSubdomains="true">internal.example.com</domain>
           <trust-anchors>
                <!-- Only trust the CAs included with the app
                     for connections to internal.example.com -->
                <certificates src="@raw/cas" />
           </trust-anchors>
      </domain-config>
 </network-security-config>
```

Trusting user-added CAs for some domains

To allow your app to trust user-added CAs for multiple domains, include something like this in your Network Security Config.

```
<network-security-config>
      <domain-config>
           <domain includeSubdomains="true">userCaDomain.com</domain>
           <domain includeSubdomains="true">otherUserCaDomain.com</domain>
           <trust-anchors>
                  <!-- Trust preinstalled CAs -->
                  <certificates src="system" />
                  <!-- Additionally trust user added CAs -->
                  <certificates src="user" />
           </trust-anchors>
      </domain-config>
 </network-security-config>
```

Trusting user-added CAs for all domains except some

To allow your app to trust user-added CAs for all domains, except for those specified, include something like this in your Network Security Config.

```
<network-security-config>
      <base-config>
           <trust-anchors>
                <!-- Trust preinstalled CAs -->
                <certificates src="system" />
                <!-- Additionally trust user added CAs -->
                <certificates src="user" />
           </trust-anchors>
      </base-config>
      <domain-config>
           <domain includeSubdomains="true">sensitive.example.com</domain>
           <trust-anchors>
                <!-- Only allow sensitive content to be exchanged
             with the real server and not any user or
    admin configured MiTMs -->
                <certificates src="system" />
           <trust-anchors>
      </domain-config>
 </network-security-config>
```

Trusting user-added CAs for all secure connections

To allow your app to trust user-added CAs for all secure connections, add this in your Network Security Config.

```
<network-security-config>
      <base-config>
            <trust-anchors>
                <!-- Trust preinstalled CAs -->
                <certificates src="system" />
                <!-- Additionally trust user added CAs -->
                <certificates src="user" />
           </trust-anchors>
      </base-config>
 </network-security-config>
```

Standardized set of system-trusted CAs

To provide a more consistent and more secure experience across the Android ecosystem, beginning with Android Nougat, compatible devices trust only the standardized system CAs maintained in AOSP.

Previously, the set of preinstalled CAs bundled with the system could vary from device to device. This could lead to compatibility issues when some devices did not include CAs that apps needed for connections as well as

potential security issues if CAs that did not meet our security requirements were included on some devices.

What if I have a CA I believe should be included on Android?

First, be sure that your CA needs to be included in the system. The preinstalled CAs are **only** for CAs that meet our security requirements because they affect the secure connections of most apps on the device. If you need to add a CA for connecting to hosts that use that CA, you should instead customize your apps and services that connect to those hosts. For more information, see the *Customizing trusted CAs* section above.

If you operate a CA that you believe should be included in Android, first complete the Mozilla CA Inclusion Process and then file a feature request against Android to have the CA added to the standardized set of system CAs.
