Una nueva campaña de malware que utiliza señuelos con temática de coronavirus para atacar a los sectores del gobierno y energía en Azerbaiyán, fue descubierta utilizando troyanos de acceso remoto (RAT) capaces de extraer documentos confidenciales, pulsaciones de teclas, contraseñas y hasta imágenes de la cámara web.

Los ataques dirigidos emplean documentos de Microsoft Word como cuentagotas para desplegar una RAT previamente desconocida basada en Python llamada *«PoetRAT»,* debido a distintas referencias a sonetos del dramaturgo inglés William Shakespeare.

> *«El RAT tiene todas las características estándar de este tipo de malware, proporcionando un control total del sistema comprometido para la operación»*, dijo Cisco Talos en un análisis.

Según los investigadores, el malware está dirigido específicamente a los sistemas de control de supervisión y adquisición de datos (SCADA) en la industria energética, como los sistemas de turbinas eólicas, cuyas identidades se desconocen en la actualidad.

El desarrollo es el último de un aumento en los ataques cibernéticos que explotan los temores continuos por la pandemia del coronavirus, como cebo para instalar malware, robar información y así obtener ganancias.

La campaña funciona agregando PoetRAT a un documento de Word, que al ser abierto, ejecuta una macro que extrae el malware y lo ejecuta.

El mecanismo de distribución exacto del documento de Word sigue sin estar claro, pero debido a que los documentos están disponibles para descargar desde una URL simple, los investigadores sospechan que las víctimas están siendo engañadas para descargar el RAT por medio de URL maliciosas o correos electrónicos de phishing.

Talos también mencionó que este ataque fue descubierto desde febrero, en algunos ataques donde se utilizaron documentos señuelos que aseguran ser de agencias gubernamentales de

Azerbaiyán y de la Organización de Investigación y Desarrollo de Defensa de India (DRDO), o aludiendo a COVID-19 en sus nombres de archivo *«C19.docx»* sin ningún contenido real.

Independientemente del vector de ataque, la macro de Visual Basic Script en el documento escribe el malware en el disco duro como un archivo llamado *«smile.zip»*, que consiste en un intérprete de python y el mismo RAT.

El script de Python también verfica el entorno donde se abre el documento para asegurarse de que no se encuentre en una sandbox, basándose en el supuesto caso de que las sandbox tienen discos duros de menos de 62 GB. Si se detecta un entorno de espacio aislado, se elimina del sistema.

El RAT cuenta con dos scripts: *«frown.py»*, que se encarga de comunicarse con un servidor remoto de comando y control (C2), con un identificador de dispositivo único y un *«smile.py»*, que maneja la ejecución de comandos C2 en la máquina comprometida.

Los comandos hacen posible que un atacante cargue archivos confidenciales, realice capturas de pantalla, finalice procesos del sistema, registre pulsaciones de teclas «Klog.exe», y robe contraseñas almacenadas en navegadores «Browdec.exe».

Además, los autores de la campaña también implementaron herramientas de explotación adicionales, como «dog.exe», un malware basado en .NET que monitorea las rutas del disco duro y transmite automáticamente la información por medio de una cuenta de correo electrónico o un FTP. Otra herramienta llamada «Bewmac», permite al atacante tomar el control de la cámara web de la víctima.

El malware gana persistencia al crear claves de registro para ejecutar el script de Python e incluso, puede realizar modificaciones en el registro para evitar la verificación de evasión de sandbox antes mencionada, con el posible fin de evitar volver a verificar el mismo entorno nuevamente.

> *«El actor monitoreó directorios específicos, señalando que querían filtrar cierta información acerca de las víctimas»,* dijeron los investigadores.

> *«El atacante no solo quería información específica obtenida de las víctimas, sino también un caché completo de información relacionada con la víctima. Al utilizar Python y otras herramientas basadas en Python durante su campaña, el hacker pudo haber evitado la detección mediante herramientas tradicionales que incluyeron Python en la lista blanca»,* agregaron.