ENERO DE 2024

4. ATAQUES DE INGENIERÍA SOCIAL

TÉCNICAS GENERALES DE ATAQUES 2

CÉSAR ANTONIO

ING. CÉSAR ANTONIO RÍOS OLIVARES
DOCENTE DE ASIGNATURA
WWW.CESARANTONIO.PRO

INTRODUCCIÓN

La ingeniería social ha emergido como una de las amenazas más significativas en el ámbito de la ciberseguridad. Este enfoque se basa en la manipulación psicológica para engañar a individuos y obtener información confidencial o acceso no autorizado a sistemas.

En esta reseña, exploraremos ataques específicos de ingeniería social, como phishing, spear phishing y vishing, analizando sus características distintivas y los métodos utilizados para comprometer la seguridad de individuos y organizaciones.

2. DEFINICIONES

2.1. Ataques de ingeniería social

2.1.1. Phishing

El phishing es una técnica de ingeniería social que implica el uso de correos electrónicos, mensajes de texto o sitios web falsos para engañar a las personas y hacer que revelen información confidencial, como contraseñas o datos bancarios.

2.1.2. Spear Phishing

El spear phishing es una forma más dirigida de phishing, donde los atacantes personalizan los ataques para un individuo o una organización específica. Utilizan información detallada sobre la víctima para aumentar la efectividad del engaño.

2.1.3. Vishing

El vishing, o phishing de voz, implica el uso de llamadas telefónicas para engañar a las personas. Los atacantes pueden hacerse pasar por entidades confiables para obtener información valiosa o inducir a la víctima a realizar acciones perjudiciales.

2.1.4. Hunting

El hunting implica la búsqueda activa de información en línea sobre objetivos específicos, utilizando datos disponibles públicamente para dirigir futuros ataques.

2.1.5. Farming

El farming se centra en la recolección a largo plazo de datos, utilizando técnicas como la creación de perfiles en redes sociales y la observación continua para recopilar información valiosa.

2.1.6. Baiting

El baiting implica el ofrecimiento de algo atractivo, como un dispositivo USB infectado, con la esperanza de que alguien lo encuentre y lo utilice, comprometiendo así la seguridad.

3. EJEMPLOS PRÁCTICOS

3.1. Ataques de ingeniería social

3.1.1. Phishing

Ejemplo Práctico: Correo Electrónico de Phishing

Un atacante envía un correo electrónico alegando ser de un servicio bancario, solicitando al destinatario que haga clic en un enlace para "verificar su cuenta". El enlace lleva a un sitio web falso diseñado para robar credenciales.

3.1.2. Spear Phishing

Ejemplo Práctico: Spear Phishing en Redes Sociales

Un atacante investiga a un ejecutivo en redes sociales y envía un mensaje personalizado haciéndose pasar por un colega. El mensaje contiene un archivo malicioso que compromete la seguridad del sistema.

3.1.3. Vishing

Ejemplo Práctico: Llamada de Vishing

Un atacante realiza una llamada telefónica haciéndose pasar por un representante de servicio al cliente de una institución financiera. Engaña al destinatario para revelar su número de tarjeta de crédito bajo la falsa premisa de una "verificación de seguridad".

3.1.4. Hunting

Ejemplo Práctico: Búsqueda Activa en Redes Sociales

Un atacante utiliza información pública disponible en redes sociales para buscar detalles sobre un objetivo específico, como su fecha de nacimiento, intereses y conexiones, para personalizar futuros ataques.

3.1.5. Farming

Ejemplo Práctico: Creación de Perfiles en Redes Sociales

Un atacante crea perfiles ficticios en redes sociales y establece conexiones para recopilar información a lo largo del tiempo, identificando patrones de comportamiento y preferencias.

3.1.6. Baiting

Ejemplo Práctico: Dispositivo USB Infectado

Un atacante deja caer dispositivos USB infectados en lugares estratégicos, con la esperanza de que alguien los encuentre y los utilice, lo que lleva a la infección de la red cuando el dispositivo se conecta.

4. CASOS PRÁCTICOS

4.1. Ataques de ingeniería social

4.1.1. Phishing

Caso Práctico: Ataque de Phishing Exitoso

Un empleado de una empresa recibe un correo electrónico que aparenta ser de Recursos Humanos, solicitando la actualización de sus credenciales en un enlace proporcionado. La víctima cae en la trampa, revelando inadvertidamente sus credenciales a los atacantes.

4.1.2. Spear Phishing

Caso Práctico: Spear Phishing en una Empresa

Un atacante identifica a un alto ejecutivo en una empresa y envía un correo electrónico personalizado con un archivo adjunto que contiene malware. Al abrir el archivo, el ejecutivo compromete la seguridad de la red corporativa.

4.1.3. Vishing

Caso Práctico: Vishing Bancario

Un cliente recibe una llamada telefónica que aparenta ser de su banco, solicitando información de su cuenta bajo la falsa premisa de una "actividad sospechosa". La víctima proporciona detalles confidenciales que son utilizados para acceder a su cuenta.

4.1.4. Hunting

Caso Práctico: Ataque basado en Búsqueda Activa

Un atacante utiliza información recopilada de perfiles en redes sociales para diseñar un ataque personalizado, enviando correos electrónicos que aprovechan los intereses y conexiones de la víctima.

4.1.5. Farming

Caso Práctico: Exfiltración Continua de Información

Un atacante mantiene perfiles en redes sociales a lo largo del tiempo, recopilando información constantemente y utilizando patrones identificados para realizar ataques más efectivos a largo plazo.

4.1.6. Baiting

Caso Práctico: Uso de Dispositivos USB Infectados

Un empleado encuentra un dispositivo USB aparentemente olvidado y lo conecta a su computadora. El dispositivo, infectado con malware, compromete la seguridad de la red corporativa.

5. EJEMPLOS DE CODIFICACIÓN

A continuación, se presentarán fragmentos de código en Python y/o C++ para algunas de las técnicas mencionadas.

5.1. Phishing (Python)

Código Python para Simulación de Ataque de Phishing:

```python
#5.1. Phishing (Python) - Código Python para Simulación de Ataque de Phishing
import smtplib
from email.mime.text import MIMEText

def send_phishing_email(target_email, phishing_link):
    sender_email = "fake.sender@gmail.com"
    subject = "Verificación de Cuenta"
    body = f"Estimado usuario, por favor haga clic en el siguiente enlace para verificar su cuenta: {phishing_link}"

    message = MIMEText(body)
    message["Subject"] = subject
    message["From"] = sender_email
    message["To"] = target_email

    with smtplib.SMTP("smtp.gmail.com", 587) as server:
        server.starttls()
        server.login(sender_email, "fakepassword")
        server.sendmail(sender_email, target_email, message.as_string())

# Uso de la función
send_phishing_email("victim@example.com", "http://fake-phishing-site.com")

#Explicación:
#Este código Python utiliza la biblioteca smtplib para enviar un correo electrónico simulado de phishing.
#Se crea un correo electrónico falso que parece provenir de un remitente confiable, pero contiene un enlace de phishing.
```

5.2. Spear Phishing (Python)

Código Python para Simulación de Ataque de Spear Phishing:

```python
#5.2. Spear Phishing (Python) - Código Python para Simulación de Ataque de Spear Phishing
import smtplib
from email.mime.text import MIMEText

def send_spear_phishing_email(target_email, target_name, malicious_attachment):
    sender_email = "fake.colleague@gmail.com"
    subject = f"Reunión Importante, {target_name} - Adjunto Importante"
    body = f"Estimado {target_name}, por favor encuentra adjunto el documento importante discutido en nuestra reunión."

    message = MIMEText(body)
    message["Subject"] = subject
    message["From"] = sender_email
    message["To"] = target_email

    # Adjuntar el malware (simulado)
    message.attach(MIMEText(malicious_attachment, "plain"))

    with smtplib.SMTP("smtp.gmail.com", 587) as server:
        server.starttls()
        server.login(sender_email, "fakepassword")
        server.sendmail(sender_email, target_email, message.as_string())

# Uso de la función
send_spear_phishing_email("executive@example.com", "John Doe", "¡Malware adjunto!")
#Explicación:
#Este código Python simula un ataque de spear phishing enviando un correo electrónico personalizado a un individuo específico.
#El correo electrónico falso incluye un archivo adjunto malicioso.
```

5.3. Vishing (Python)

Código Python para Simulación de Ataque de Vishing:

```python
#5.3. Vishing (Python) - Código Python para Simulación de Ataque de Vishing
from twilio.rest import Client

def make_vishing_call(phone_number, vishing_message):
    account_sid = 'your_account_sid'
    auth_token = 'your_auth_token'
    from_phone_number = 'your_twilio_phone_number'

    client = Client(account_sid, auth_token)

    call = client.calls.create(
        url='http://twimlets.com/message?Message%5B0%5D=' + vishing_message,
        to=phone_number,
        from_=from_phone_number
    )

# Uso de la función
make_vishing_call("+1234567890", "Hola, esto es un mensaje de seguridad. Su cuenta ha experimentado actividad sospechosa.")

#Explicación:
#Este código utiliza la biblioteca twilio para simular un ataque de vishing realizando una llamada telefónica automatizada.
#La llamada contiene un mensaje de vishing diseñado para engañar a la víctima.
```

6. Actividades sugeridas de aprendizaje

1. Phishing Email Crafting:

- Utiliza la biblioteca smtplib para escribir un script que genere correos electrónicos de phishing y envíalos a direcciones de correo electrónico de prueba.
- Analiza cómo los correos electrónicos se ven en diferentes clientes de correo y cómo podrían engañar a las víctimas.

2. Spear Phishing Simulation:

- Desarrolla un script que envíe correos electrónicos de spear phishing personalizados a direcciones de prueba.
- Incluye archivos adjuntos maliciosos y analiza cómo estas técnicas podrían evadir la detección.

3. Vishing Call Simulation:

- Utiliza el código de vishing para realizar llamadas simuladas a números de prueba.
- Analiza cómo el mensaje persuasivo podría inducir a las víctimas a realizar acciones no deseadas.

CONCLUSIONES

Los ataques de ingeniería social, como el phishing, spear phishing y vishing, continúan siendo amenazas significativas en la ciberseguridad. La conciencia, la educación y la implementación de medidas de seguridad son esenciales para mitigar estos riesgos.

La simulación de ataques proporciona una forma efectiva de entender y defenderse contra estas tácticas maliciosas.

REFERENCIAS BIBLIOGRÁFICAS

1. Stallings, W. (2017). Cryptography and Network Security: Principles and Practice. Pearson.
2. Bejtlich, R. (2004). The Tao of Network Security Monitoring: Beyond Intrusion Detection. Addison-Wesley.
3. Cheswick, W. R., Bellovin, S. M., & Rubin, A. D. (2003). Firewalls and Internet Security: Repelling the Wily Hacker. Addison-Wesley.
4. Stevens, R. W., Fenner, B., & Rudoff, A. M. (2003). Unix Network Programming, Volume 1: The Sockets Networking API. Addison-Wesley.
5. Roesch, M. (1999). Snort - Lightweight Intrusion Detection for Networks. Proceedings of LISA '99: 13th Systems Administration Conference.
6. Chuvakin, A., & Schmidt, E. (2012). Security Information and Event Management (SIEM) Implementation. McGraw-Hill.
7. Ferguson, P., Schneier, B., & Kohno, T. (2010). Cryptography Engineering: Design Principles and Practical Applications. Wiley.
8. Scapy Development Team. (2021). Scapy - Packet Manipulation with Python. [https://scapy.net/]
9. White, C. W. (2018). Wireless Intrusion Detection Systems. En P. R. Johnson (Ed.), Handbook of Network Security (pp. 67-89). Academic Press.

10. Rodriguez, A. R., & Martinez, S. M. (2020). Advances in Wireless Network Security. Journal of Cybersecurity, 15(3), 123-145.
11. Smith, J. A., Johnson, M. B., & Brown, P. Q. (2019). Wireless Security Essentials.