Neusoft东软

NetEye Event Language
技术白皮书

沈阳东软软件股份有限公司

2006 年 2 月

目 录

第一章 NEL 简介及基于 NEL 的层次化开发模式

NEL（NetEye Event Language）是东软自主研发成功的一种通用攻击描述语言，包括 NEL 语言规范及相应的开发环境。NEL 的设计目标是为不同类型的入侵检测防御产品（如网络 IDS/IPS、主机 IDS/IPS、应用防火墙、应用安全增强模块等）的开发提供一个具有强大描述能力、高度的可扩展性和很高代码执行效率的语言平台。NEL 开发环境（以下简称 NEL 平台）可以大大提高协议分析和攻击检测防御规则开发的效率，极大地降低开发团队的沟通成本和代码维护代价。同时 NEL 平台提供了无缝使用 C 语言中的常量、变量、结构、函数等语言实体的能力，从而可以复用大量已有的 C 程序代码，有效缩短产品的面世时间以及更新升级周期。

NEL 是为了解决大型攻击检测防御系统开发过程中所面临的难题而提出的。为了准确高效地描述、检测和防御攻击，攻击检测防御系统通常采用两种模式来进行开发：

1 全部使用通用高级语言（比如 C）来实现协议分析和攻击检测。这种开发模式的问题在于，每当新的攻击出现，开发者增加相应的规则后就需要重新编译系统，而且需要开发人员对于攻击有深入的了解，系统的可扩展性和可维护性都比较差。对于大型的攻击检测防御系统来说，这种开发模式基本上是不可行的。

2 使用专用的攻击描述语言进行开发。为了解决第一种模式存在的问题，很多 IDS/ IPS 厂商以及研究机构提出了一些攻击描述语言，比如 SourceFire 公司的 SNORT 语言、NFR 的 N-Code 语言、V. Paxson 提出的 Bro 语言等。采用这些语言开发的系统，在增加新的规则后无须重新编译系统，从而大大缩短了攻击应对的时间。但这些语言都将协议分析固化到语言本身当中，当需要增加一个新的协议防护模块时，就必须对语言本身进行扩充。由于大型攻击检测防御系统往往需要几十人、甚至是上百人开发并维护数百种协议防护模块，数千条攻击检测防御规则，要求每个开发人员都具有扩充语言平台的能力是不可能的。这种限制使得基于上述语言平台开发的攻击检测防御产品的功能扩展性受到了极大的制约。

NEL 平台有效解决了上述两种开发模式所面临的难题。基于 NEL 平台，可以将基于协议

分析的攻击检测防御任务划分为：协议分析的开发，攻击检测防御规则的开发和语言平台的开发这三个子任务，每个子任务分别由协议分析小组、攻击分析小组、攻击描述语言开发小组来承担。这样，即使系统增加了很多的协议、制订了很多条攻击检测防御规则、描述语言的语法不断增强，每个子系统的扩展以及规模的不断增长不会影响到其他开发小组中的开发人员，极大地提升了大规模攻击检测防御系统的开发效率。

采用 NEL 平台不仅解决了分层次开发和系统扩展性的难题，而且由于 NEL 在运行期将“协议分析”和“攻击检测”紧密耦合在一起，因此开发出的系统具有非常高的运行效率。NEL 平台全新开发模式以及在代码执行速度等方面的诸多技术优势使得它成为一个理想的高性能攻击检测防御产品的开发平台。

第二章 NEL 语言的技术优势

2.1 强大的攻击描述能力

NEL 是一种过程型编程语言，提供了很多高级语言中才有的过程性手段，具有强大的描述复杂攻击的能力。NEL 中可以定义各种数据类型、常量、变量、数组、表达式、函数等等，这些特性使得 NEL 达到了与 C 语言相当的描述能力。同时，NEL 的语法与 C 非常相近，在 NEL 中可以定义一些抽象的语言实体（变量、函数），然后以过程性的方式操作这些语言实体来完成运算。

NEL 引入了一个新的语言元素－事件。事件指开发者或 NEL 语言本身定义的攻击检测防御过程中检测到的活动：检测到一个 TCP 级的数据报/一个 HTTP 请求/一次 SMTP 通信等等都可以被定义为一个事件。针对不同的应用协议，开发人员可以采用 NEL 中的事件在任何层次上（数据包层次、协议层次以及更高的层次上）来定义事件并基于事件来制订攻击检测防御规则。比如，我们可以采用 HTTP_REQ_GET 代表系统收到 HTTP GET 请求这一事件；在此基础上，采用 HTTP_REQ_GET_ABNORMAL 代表所有异常的不符合 HTTP 协议规范的 GET 请求（在 HTTP_REQ_GET 事件中通过限定条件来检测，这些限定条件可以基于协议数据包格式规范和协议交互过程规范来设定）；也可以通过 HTTP_REQ_GET_DDOS 来定义利用 HTTP GET 请求发起的 DDOS 攻击（通过对 HTTP_REQ_GET 事件按照 IP 和时间间隔进行统计来检测）。从以上例子可以看出，NEL 中事件的引入为开发人员制订攻击检测防御规则提供了极大的灵活性。

NEL 中的“事件”概念使得在攻击检测防御系统的开发过程中，协议分析开发人员只要专注于“基础协议事件的定义和产生”，协议分析的开发完成后，协议分析的知识即被固化到系统中；攻击检测防御规则开发人员可以专注于“攻击事件的定义”，而无需知道协议分析的细节。事件的引入使得协议分析的开发和攻击检测防御规则的开发被划分成两个完全独立的层

次。举一个简单的例子：利用 SMTP EHLO 请求可以构造出若干种攻击，我们可以在 NEL 中定义 SMTP_EHLO 代表系统（IDS/IPS 等）检测到 SMTP 的 EHLO 请求这一事件，完成 SMTP_EHLO 事件的协议检测代码后，有关 SMTP EHLO 请求的分析检测的知识即被固化在系统中，再进行攻击检测防御规则的开发时，开发人员不再需要了解 SMTP_EHLO 事件的协议分析过程，只需在编写规则时直接使用 SMTP_EHLO 事件及事件的各种属性（如长度，内容等）即可，攻击检测防御规则的代码非常简洁，同时也保证了系统具有很高的处理性能。对于没有将协议分析开发和攻击检测防御规则开发划分成两个层次的语言，开发人员在编写利用 SMTP EHLO 请求发起的攻击的检测防御规则时，必须对协议细节有深入了解，而且协议的分析过程必须体现在每一条规则中，不仅代码变得非常庞杂，系统运行时的效率也会受到非常大的影响。

NEL 不仅引入了“事件”这一概念，还引入了对事件自身和事件之间的逻辑约束关系，比如对事件进行限定和归结等操作。比如，用 HTTP_REQ 代表系统接收到一个 HTTP 请求；则 HTTP_REQ_GET :HTTP_REQ($1->method==HTTP_GET)代表对 HTTP_REQ 进行限定后得到的一个更细粒度的事件，即系统接受到一个 HTTP GET 的请求； 进一步还可以限定出 HTTP_REQ_GET_LONG: HTTP_REQ_GET ($1->len >1024)，指的是系统接收到一个长度大于 1024 字节的 HTTP GET 请求。以上是事件限定的例子，再来看一个事件归结的例子：在检测 SMTP 攻击时，针对客户端发给服务器的请求，协议开发者可以定义 SMTP_EHLO/SMTP_RCPT/SMTP_MAIL 等事件，同时，所有这些事件还可以归结成一个 SMTP_REQ 事件，表示为：

SMTP_REQ：SMTP_EHLO | SMTP_RCPT | SMTP_MAIL | …

事件和事件逻辑约束关系的引入使得 NEL 具有了强大的攻击描述能力，同时提供了攻击检测防御所必须的抽象性和概括性，使得 NEL 开发者可以方便地编写基于协议异常、漏洞特征等不同类型的攻击检测防御规则，由于规则具有对协议上下文的精确理解，包括了漏洞特征的描述，因此具有非常高的检测准确性。

2.2 良好的可扩展性

NEL 平台及 NEL 平台之上的协议分析模块和攻击检测防御规则都可以不断地扩展：NEL 平台自身的扩展可以为开发者提供越来越强大的协议和攻击描述、分析和防御能力，使得攻击检测具有越来越高的准确率和性能，而这种扩展对平台上已经开发出的协议分析模块和攻击检测防御规则不会有任何影响；同样，开发者也可以不断进行协议分析模块的扩充，以提供更强的针对特定协议的攻击检测防御能力，而这种扩充对于协议之上已有的攻击检测防御规则也不会有任何影响。

在现实世界中，许多协议都是非常复杂的，受时间和人力投入的制约，在攻击检测防御系统的开发过程中，一开始就将协议所有的命令和交互过程都实现是非常困难的，也是不必要的。NEL 提供了一种机制，允许开发者将一个协议所有未做全面分析的命令作为一个统一的事件来处理，并基于这个事件来定义攻击检测防御规则，从而使得协议分析的开发成为一个渐进的过程，开发重点和次序可以根据攻击的危害程度和影响范围、人力资源状况等多方面因素灵活调整。这种渐进式的协议分析开发模式对于应对复杂多变的网络攻击是十分必要的。

为了有效利用已有的 C 语言代码，NEL 平台提供了无缝使用 C 语言实体的功能，开发者可以在 NEL 中无缝地调用 C 语言的代码。在这种模式下，C 语言是一个宿主语言，而 NEL 是一个寄生语言。采用 NEL 构建攻击检测防御系统，可以有效复用大量已有 C 代码，迅速增强和扩展系统的攻击防御能力。

NEL 平台高度的可扩展性使得采用 NEL 开发的产品可以快速开发出针对新的漏洞和攻击的检测防御规则和防护方案，消除随着被发现的漏洞数目不断增加而日益增长的安全风险。

2.3 高效的分析检测过程

NEL 的协议分析和攻击检测防御过程在运行期是紧耦合在一起的，因此具有非常高的运行效率，参见图 1：

系统顺序接收到 4 个数据包，即发生了 4 个事件 t0，t1，t2，t3；

如果第一个事件 t0 满足约束条件 P1，则我们得到了一个 REQ 事件；

如 REQ 事件满足约束条件 P3，则我们得到了一个 REQ［P3］事件；

如果第二个事件 t1 满足约束条件 P2，则我们得到了一个 ACK 事件；

如果 ACK 事件满足约束条件 P4，则我们得到一个 ACK［P4］事件；

如果 REQ 和 ACK 两个事件满足一定的时序关系（在这个例子中，是 REQ 在前，且其后紧跟 ACK 事件），则我们得到了一个 RA 事件；

如果 REQ [P3]和 ACK [P4]两个事件满足一定的时序关系（在这个例子中，是 REQ［P3］在前，且其后紧跟 ACK［P4］事件），则我们发现了攻击事件 ATK，同时进行相应的处理。

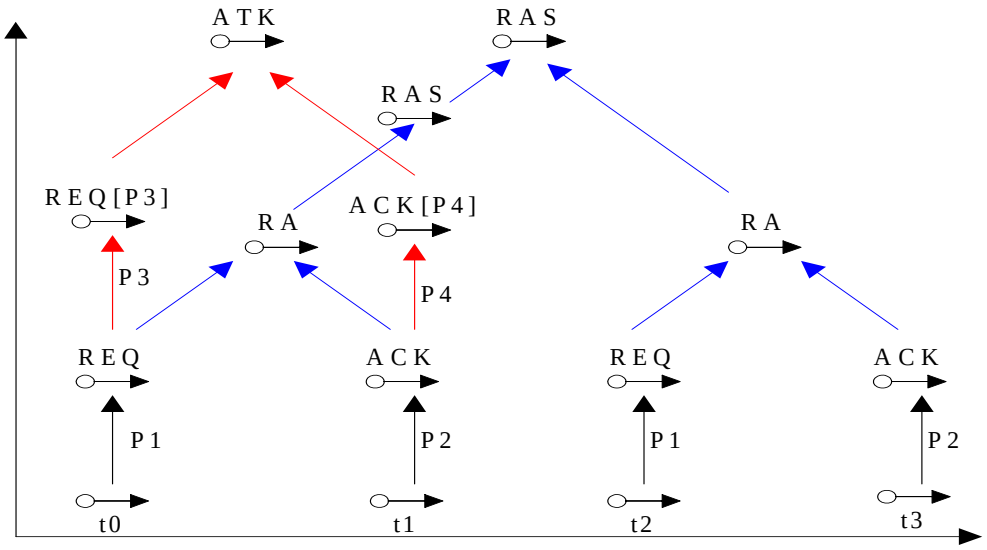

图 1　NEL 协议分析及攻击检测过程示例

图 1 从横向上看，反映的是随着时间推移对网络事件和攻击的分析过程。以协议分析过程为例，协议状态的转换是受协议状态机约束的，这种约束可以通过 NEL 规则体现出来，从而在分析过程中可以根据当前的协议状态有效减少分析的计算量。比如，在上图中，当事件 t1 发生时，我们只要看一下它是不是满足约束条件 P2 即可，而无需判断它是否满足约束条件 P1。高效的事件分析过程保证了系统的运行效率。

图 1 从纵向上看，反映的是事件不断抽象的过程和层次性结构。原始事件（t0,t1,t2,t3 等）

满足一定的约束条件后，会形成一个抽象事件，而后在更高层次的分析过程中，将只针对高层的抽象事件进行，这大大降低了攻击检测防御过程所需匹配的规则数量。

由于 NEL 具有在横向和纵向上减少匹配规则的能力，因此采用 NEL 开发的攻击检测防御系统在检测效率和检测性能上远远优于 Snort 那样的系统。

下表是 NEL 与 SNORT、BRO 这两种目前比较流行的攻击描述语言的对比。

		NEL	SNORT	BRO
开发模式		语言平台/协议分析/攻击检测防御规则的开发可分别由三个小组分别完成，层次清晰，彼此的工作不会互相影响	语言平台与攻击检测防御规则的开发可由两个小组分别完成，彼此的工作不会相互影响；协议分析与攻击检测防御规则紧藕合在一起，检测规则中必须包含协议分析过程	语言平台与攻击检测防御规则的开发可由两个小组分别完成，彼此的工作不会相互影响； 语言平台与协议分析紧藕合在一起，必须由同一个小组完成
规则开发人员的要求		无需了解协议分析和语言平台的内部细节	需要了解协议分析的内部细节	需要了解语言平台的内部细节
协议扩展性		强	弱	强
检测准确性		高	低	高
检测能力	协议分析能力	强	弱	强
	漏洞描述能力	强	弱	强
	抵御变形攻击能力	强	弱	强
运行效率		编译模式，运行效率高	编译模式，运行效率高	解释模式，运行效率低
利用已有 C 代码的能力		具备	不具备	不具备

表 1　NEL 与其它攻击描述语言的对比

对于攻击检测防御系统开发来说，攻击描述语言的检测准确性、扩展性、运行效率以及开发模式等方面的因素对于产品的成功是至关重要的。通过上表对比可以看出，NEL 在攻击描述能力、检测准确性、运行效率和开发效率等诸多方面都具有明显优势，非常适合于应用于高速网络环境中的深度防御产品（如 IPS、应用防火墙等）的开发。

第三章 采用 NEL 编写攻击检测防御规则

开发者可以采用 NEL 基于协议异常、漏洞特征或攻击特征等来定义攻击检测防御规则。基于协议异常或漏洞的 NEL 规则更准确地描述了攻击的本质特征，因此具有极强的攻击抽象力和检测效率，往往一条规则就可以检测数十种攻击，并有效阻止 0-day 攻击。下面以 SMTP 协议为例，简单介绍一下如何采用 NEL 编写攻击检测防御规则：

很多攻击将发送超长的协议数据包作为一种入侵手段，因此，通过检查协议数据包长度可以抵御很多攻击，下面就是一个这样的例子。

示例 1：

```
BAD_LONG_EHLO: ehlo_req( $1-> host_name_len >= 512 )
{
    smtp_deny($0, "found a long ehlo command! len =%d\n", $1->len );
}
;
```

说明：这是一条非常简单的规则，它规定系统在检测到一个“EHLO”请求事件时，如果该请求事件的数据长度（host_name_len）大于 512 字节时，就关闭这条 SMTP 连接。这条规则可以阻断 CAN-1999-0098, CAN-1999-0284, CAN-1999-1529, CVE-2000-0042, CVE-2000-0488, CVE-2000-0507, CAN-2000-0657, CAN-2003-0264, CAN-2004-1291 等攻击。在 NEL 中，$0 通常代表一个连接 （本例中是 SMTP 连接），$1 代表规则右侧（冒号之后的部分）的第一个事件（本例中是一个 ehlo_req 事件，即系统接收到一个 EHLO 请求），依此类推。详细说明参见《NEL 事件描述语言用户手册》。

RFC 中对各种协议的状态转换、交互过程都是有严格限定和规范的，如果协议的交互过程偏离了协议规范，就是一种协议异常，而这往往意味着攻击。下面是一个通过检查协议状

态转换异常来检测抵御攻击的例子。

示例 2：

```
BAD_DATA_REQ: data_req($0->mail_state < SMTP_MAIL_STATE_RCPT_TO )
{
    smtp_deny($0, "Haven't seen RCPT TO command before this DATA command!\n");
}
;
```

说明：根据 RFC 2821，在一个合法的 SMTP 连接过程中，只有在客户端至少发送过一个“RCPT TO”请求命令之后，客户端发送“DATA”请求命令才是合法的，*$0->mail_state < SMTP_MAIL_STATE_RCPT_TO* 即代表着在 SMTP 连接过程中，客户端尚未发送过“RCPT TO”请求命令，因此通过这条规则可以阻挡住所有不符合 SMTP 协议标准交互过程的“DATA”请求命令，同时也将所有利用直接发送“DATA”请求命令进行的攻击拦截下来。

通过检查协议状态转换过程中的命令携带参数的异常，也可以有效抵御网络入侵与攻击。下面是一个例子。

示例 3：

```
BAD_MAIL_FROM_REQ: mail_from_req( $1->key == SMTP_MAIL_SIZE && $0->cmd_allow[SMTP_ALLOW_SIZE] == 0 )
{
smtp_deny($0, "found a MAIL FROM command with keyword SIZE without declaration of SIZE at previous EHLO ACK!\n");
}
;
```

说明：当 SMTP 服务器对客户端的 EHLO 请求应答时，同时会规定后续过程中允许客户端发向服务器端的命令所携带的参数，$0->cmd_allow[SMTP_ALLOW_SIZE]==0 代表不允许客户端后续发送的 MAIL_FROM 命令中携带 SIZE 参数，那么如果后续客户端发送的 MAIL

FROM 请求携带了 SIZE 参数（即$1->key == SMTP_MAIL_SIZE），则出现了协议异常情况，说明请求可能来自一个试图入侵 SMTP 服务器的恶意攻击者，此连接将被阻断掉。

示例 4：

```
event struct smtp_cmd_vrfy VRFY_SHELL_CODE;
BAD_VRFY_REQ:VRFY_SHELL_CODE(nel_frag_match($1->mail_box,
$1->mail_box_len,   "\x83\xc1\x0A\xFF\xE1" ))
{
    smtp_deny($0, "Found a C-Mail SMTP Server Remote Buffer Overflow Exploit!\n");
    return 0;
}
;
event struct smtp_cmd_vrfy VRFY_SHELL_CODE;
VRFY_SHELL_CODE: vrfy_req( nel_has_shellcode($1->mail_box, $1->mail_box_len))
{
    print("Found a shell code in VRFY command!\n");
    return $1;
}
;
```

说明：以上规则是一个检测 C-Mail SMTP Server 远程缓冲区溢出攻击的例子。对于 vrfy_req 事件，系统可以调用缓冲区溢出检测函数 nel_has_shellcode 检测协议数据包中是否包含缓冲区溢出，如果判断结果为是，它将形成一个 VRFY_SHELL_CODE 事件，即在 vrfy_req 事件中发现了缓冲区溢出攻击。

在此基础之上，如果该 VRFY_SHELL_CODE 事件还能满足一定的特征，即可以匹配一个特定的字符串（"\x83\xc1\x0A\xFF\xE1" ），那么它将形成一个 BAD_VRFY_REQ 事件，

即系统在 vrfy_req 事件发现了一种特定类型的缓冲区溢出攻击－C-Mail SMTP Server 远程缓冲区溢出攻击，同时连接将被阻断掉。

示例 4 说明，如果开发者不仅了解漏洞的特征，还了解利用这一漏洞的某个攻击的具体特征，那么可以同时基于漏洞特征和攻击特征定义 NEL 检测防御规则。基于漏洞特征和基于攻击特征的规则相结合，可以了解更多攻击者所使用的攻击方法的信息，更准确地判断安全事件的危害程度并追查攻击。

第四章 NEL 应用领域及 NEL 开发联盟

由于 NEL 平台具有强大的攻击描述能力、优异的可扩展性和卓越的运行效率，同时具有无缝调用 C 语言实体的能力，因此基于 NEL 平台可以针对不同的安全需求开发相应的安全产品，NEL 平台的主要应用领域包括：

1. 专注网络安全领域的厂商、科研院所和组织机构等，可以采用 NEL 平台开发 IDS/IPS、应用防火墙、蠕虫检测系统、垃圾邮件防护系统、DOS/DDOS 防护系统、异常流量监控系统、P2P 应用（BT、MSN、QQ、SKYPE 等）控制/过滤系统、安全服务器（如 WEB/MAIL/FTP/ DNS/文件服务器）等等。
2. 操作系统开发商可以采用 NEL 平台对操作系统常用的服务和协议进行安全性增强，开发安全操作系统。
3. 数据库及数据库应用开发商可以采用 NEL 平台增强数据库通信协议的安全性，加强对数据库通信过程的审查，及时发现对数据库的非法访问与恶意攻击，保障数据库自身的安全性。
4. 应用系统开发商可以采用 NEL 平台对应用（如 web 访问、web service 服务等）流量进行过滤和检查，及时发现利用应用服务进行的攻击、窃密等行为。
5. 网络设备供应商可以采用 NEL 平台增强路由协议、SIP 协议、H.248 协议的安全性，检测针对语音/视频服务系统的攻击，增强电信运营网络的安全性。
6. 移动软件开发商可以采用 NEL 平台开发手机等移动设备的安全模块，检测手机病毒等新的威胁。

基于 NEL 的安全产品具有非常强的客户化和定制化能力。以 Web 应用为例，通过在规则中引入应用环境相关的信息（如 Web 应用所允许使用的参数数量、参数名、数据类型、页面语言、通信协议等），可以开发出定制化的 NEL 规则，增强其适应性和准确性，更好地抵御各类缓冲区溢出攻击、SQL 注入、URL 攻击等等。

NEL 作为一种具有中国自主知识产权的攻击描述语言，东软希望 NEL 能够在信息安全领域得到广泛应用，推动中国自主知识产权的安全技术和产业的发展。为此，东软推动成立了“NEL 开发联盟”。作为一个知识互享和经验交流的平台，国内外信息安全领域的厂商、组织机构、技术人员可以通过“NEL 开发联盟”随时了解 NEL 平台最新的功能特性、技术进展、应用领域和发展方向，合作开展基于 NEL 平台的安全技术和产品的研究开发，共同推动“NEL”这一中国自主知识产权的攻击描述语言的应用与发展，将 NEL 打造成为象 JAVA 在企业级计算领域一样的，在网络安全计算领域“遍在”的语言平台。东软欢迎所有对 NEL 语言及 NEL 开发平台感兴趣的厂商、研究机构和开发人员加入到“NEL 开发联盟”中来。

加入“NEL 开发联盟”，您将可以：

- 获取东软 NEL 开发平台最新版本的软件、技术文档、开发指南、示例程序等信息
- 获取 NEL 开发平台最新的应用领域、成功案例及市场宣传资料
- 了解最新 NEL 技术研讨会主题及安排，并应邀参加由业界专家主持的技术研讨

“NEL 开发联盟”设立专门站点以便于会员之间相互交流，集思广益。同时设立专用邮箱：nel_alliance@neusoft.com，会员可以用电话、邮件等方式向 NEL 联盟提出各种有价值的建议、方案和设想。

如您希望了解更多NEL技术和“NEL开发联盟”的相关信息，可以访问东软NetEye网站，http://neteye.neusoft.com，点击“NEL开发联盟”栏目，获取您感兴趣的信息。

期待您的关注与加盟！